

# 68'

## MICRO JOURNAL

Australia A \$4.75 New Zealand NZ \$6.50  
Singapore S \$9.45 Hong Kong H \$23.50  
Malaysia M \$9.45 Sweden 30-SEK

**\$2.95 USA**

**Motorola** S-50 BUS•VME•MACINTOSH  
& Other 68XXX Systems  
6809 68008 68000 68010 68020 68030  
OS-9 The Magazine for Motorola CPU Devices FLEX  
For Over a Decade! SK\*DOS  
A User Contributed Journal

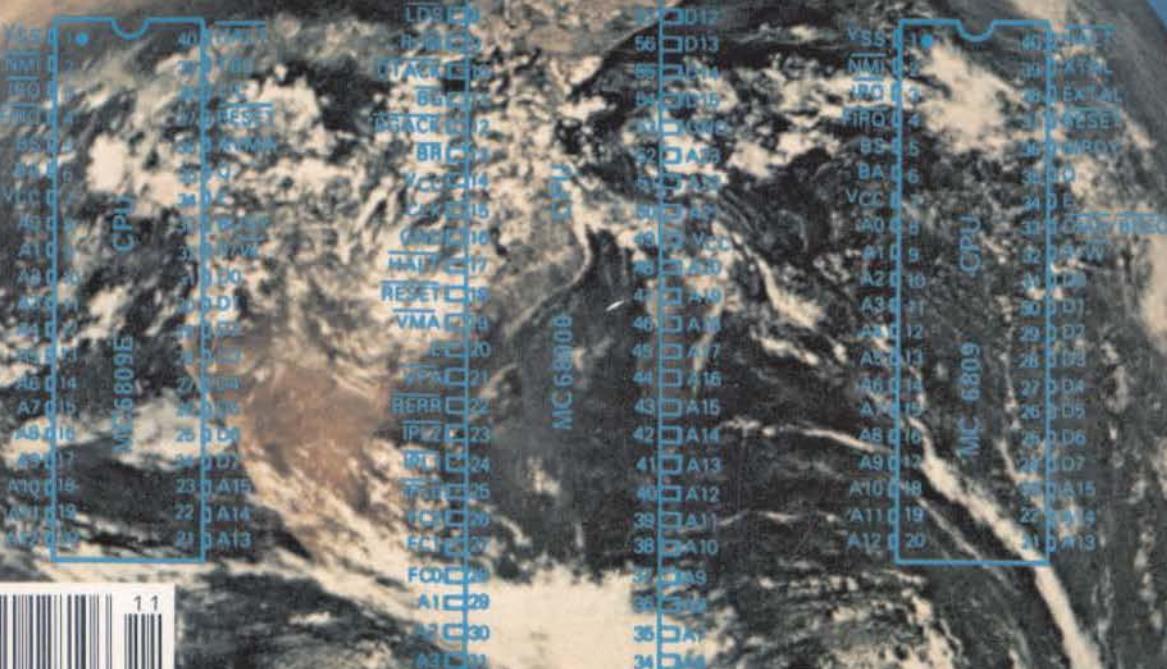
This Issue:

68000 Disk Repair p.25  
BTREE p. 27  
"C" User Notes p.19  
Basically OS-9 p.16  
Software User Notes p.8  
Drives Not Ready p.37

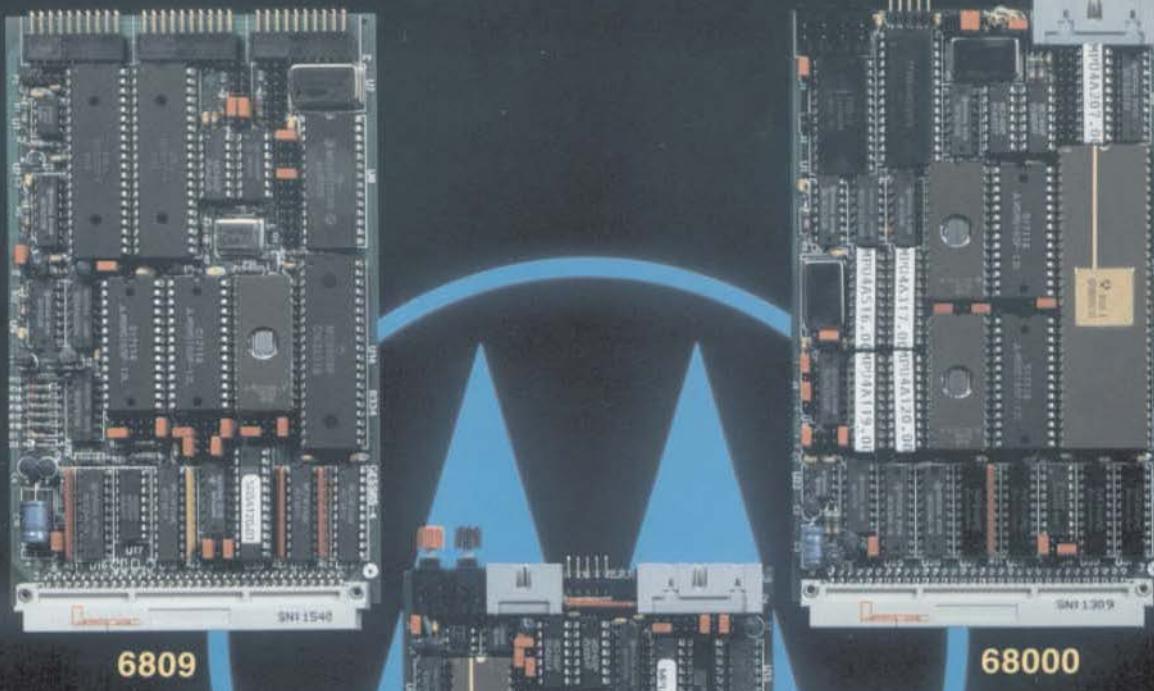
And Lots More!

**VOLUME VIII ISSUE XI • Devoted to the 68XX User • November 1986**  
"Small Computers Doing Big Things"

SERVING THE 68XX USER WORLDWIDE



# WE HAVE MOTOROLA COVERED



ON THE  BUS

## GESSBS-4 \$316\*

1 MHz 6809 CPU  
Sockets for up to 32 Kbytes EPROM  
Sockets for up to 16 Kbytes CMOS RAM  
One RS 232 serial port  
40 TTL Bidirectional I/O lines  
4 x 16-bit timers

## GESMPU-14 \$636\*

8 MHz 68010 CPU  
Optional 32081 arithmetic unit  
Sockets for up to 128 Kbytes EPROM  
One RS 232 serial port  
4 x 8-bit timers  
Real-time clock/calendar and battery

## GESMPU-4A \$316\*

8 MHz 68000 CPU  
Sockets for up to 128 Kbytes EPROM  
Sockets for up to 64 Kbytes CMOS RAM  
One RS 232 serial port  
Three 16-bit timers

## SOFTWARE

OS-9®, PDOS®, CP/M 68K®, Editor-Assembler, Basic-Pascal-C compilers, FORTH.



### USA - CANADA

100 West Hoover Ave.  
Mesa, AZ 85202  
Tel. (602) 962-5559  
Telex 386575

### INTERNATIONAL

3,chemin des Aulx  
CH-1228 Geneva  
Tel. (022) 713400  
Telex 429989

\* 100 piece quantities

 is a registered  
trademark of Motorola Inc.

# BOARDS & PARTS FOR GIMIX SYSTEMS

CONTACT GIMIX FOR PRICES, DETAILS, AND REQUIREMENTS FOR HARD DISK AND OTHER MASS STORAGE UPGRADES.

THE GIMIX CLASSY CHASSIS #19 consists of a heavyweight aluminum cabinet, constant voltage ferro resonant power supply, and S550 Mother Board with baud rate generator board	\$1498.19
#72 Triple Disk Regulator Card	\$88.22
#93 Baud Rate Generator Board	\$88.93
#23 Missing Cycle Detector	\$36.23
#92 Mother Plate	\$14.92 50 Hz Option
Cable sets 8" with Back Panel connector	\$30.00
for two 8" external drives	\$29.25
for two 5" external drives	\$44.26 for two 5" drives
	\$34.96

## CPU BOARDS

#01 GMX III CPU & OS-9 GMX III	\$1698.01
#02 GMX III CPU & UNIFLEX III	\$1998.02
The #05 GIMIX 6809 PLUS CPU Board	\$570.05
Options: GIMIX DAT	\$35.00 9511A
SWIP DAI	\$15.00 9512
803 6800 CPU	\$224.03
805 6800 CPU w/ Timers	\$288.06
6800 Baud Rate Option	Add \$30.00

## FLOPPY DISK CONTROLLER

#68 DMA	\$588.88
---------	----------

## MEMORY BOARDS FOR 6809/68020 SYSTEMS

#72 256KB CMOS STATIC RAM Board with battery back up (specify system)	\$640.72
--	----------

## MEMORY BOARDS (6800/6809 SYSTEMS ONLY)

#34 8K PROM Card	\$98.34
#32 16 Socket PROM/ROM/RAM Board, 24 pin	\$238.32
#31 16 Socket Universal Memory Board, 24/28 pin	\$268.31

## INTELLIGENT I/O PROCESSOR BOARDS

significantly reduce systems overhead by handling routine I/O functions, freeing the host CPU for running user programs. This improves overall system performance and allows user terminals to be run at up to 19.7 K baud. For use with GMX III and O20 systems.	
#11 3 Port Serial, 30 Pin (OS9)	\$498.11
#12 3 Port Serial, 30 Pin (UNIFLEX)	\$498.14
#12 Parallel, 50 Pin (UNIFLEX-O20)	\$538.12
#13 4 Port Serial, 50 Pin (OS9 & UNIFLEX-O20)	\$818.13
#15 24K Version of #11, with either large input or output buffers (specify)	\$648.15

## I/O BOARDS (6800/6809 SYSTEMS ONLY)

#41 Serial, 1 Port	\$88.41
#43 Serial, 2 Port	\$128.43
#46 Serial, 8 Port (OS9/FLEX only)	\$318.46
#42 Parallel, 2 Port	\$88.42
#44 Parallel, 2 Port (Centronics pinout)	\$128.44
#45 Parallel, 8 Port (OS9/FLEX only)	\$198.45
#50 I/O for RS-232C, 423, 422-w/6850	\$244.50
#52 SSDA with 6852	\$254.52
#54 ADLC with 6854	\$260.54

## CABLES FOR I/O BOARDS — SPECIFY BOARD

#95 Cable sets (1 needed per port)	\$24.95
#51 Cent. B/P. Cable for #12 & #44	\$34.51
#53 Cent. Cable Set	\$36.53

## OTHER BOARDS & PARTS

#66 Prototyping Board-50 Pin	\$56.68
#33 Prototyping Board-30 Pin	\$38.33
Windlatch EPROM Programmer S30 (OS9/FLEX 6809 only)	\$545.00
#76 Video Board-80 x 24	\$398.76
#88 Relay Driver Package	\$1128.08
#86 Above without Relays	\$530.86
Oplo Board	\$248.05
Blinder, 3"	\$12.00
Blinder, 2"	\$9.00

## 8" DRIVE CABINET & PARTS

2 8" DS00 Drives, Cabinet & Cables 60 Hz only	\$1698.88
Cabinet Only for 8" Drive	\$848.10
220v/50 Hz. Option Add	\$30.00
Cable Set-Internal for 2 Drives	\$44.82
Cable Set-Internal for 4 Drives	\$67.84
Cable from 8" Cab. to Mainframe	\$45.81
8" Filter Plate	\$14.83

## SOFTWARE

GIMIX exclusive versions of OS-9/GMX I, II, III & FLEX are for GIMIX hardware only. All versions of OS-9 require the #68 controller. When ordered with controller, FLEX's	\$30.00
GIMIX versions of FLEX	\$90.00
GMX VDISK for FLEX 09	\$100.00
GMXBUG: PROMs & Manual	\$148.65
Boot or Video/Boot PROMS (6809)	\$30.00
GMIX Boot PROM for UNIFLEX	\$50.00
RMS (OS9)	\$250.00
DO (OS9)	\$70.00
OS-9 GMX III Update w/CPU SPPTROM	\$125.00
I/O PROMS w/Update	\$40.00
GMXBUG/FLEX/VDISK w/OS-9 III update	Add \$175.00
RAM Disk for OS-9	\$125.00
O-FLEX	\$250.00
OS9GMX I	\$250.00
OS9 GMX II	\$500.00
SCULPTOR-6809 (UNIFLEX/OS-9)	\$995.00
SCULPTOR-68020 (UNIFLEX)	\$1595.00

CONTACT GIMIX FOR PRICES AND AVAILABILITY OF OPTIONAL UNIFLEX AND OS9 LANGUAGES AND OTHER SOFTWARE.

ALL PRICES ARE F.O.B. CHICAGO.

GIMIX DOES NOT GUARANTEE PERFORMANCE OF ANY GIMIX SYSTEMS, BOARDS OR SOFTWARE WHEN USED WITH OTHER MANUFACTURERS PRODUCT.

## LIMITED WARRANTY

GNX INC. ("GNX") WARRANTS ITS PRODUCTS AGAINST DEFECTS IN MATERIAL AND WORKMANSHIP FOR A PERIOD OF NINETY DAYS FROM THE DATE OF SHIPMENT. THE OBLIGATION OF GNX IS LIMITED TO THE REPAIR OR REPLACEMENT OF ANY PRODUCT, FREE OF ALL CHARGES, WHICH PROVES DEFECTIVE DURING THIS PERIOD. THIS WARRANTY DOES NOT COVER DAMAGE DUE TO ACCIDENTS, NEGLIGENCE, ABUSE OR TAMPERING.

GNX MAKES NO OTHER WARRANTIES OR GUARANTEES, EXPRESS, STATUTORY, OR IMPLIED, OF ANY KIND WHATSOEVER WITH RESPECT TO ANY PRODUCT PURCHASED, AND ANY IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE IS HEREBY DISCLAIMED BY GNX AND EXCLUDED FROM ANY AGREEMENT MADE BY GNX.

GNX WILL NOT BE RESPONSIBLE FOR ANY DAMAGE OF ANY KIND NOT COVERED BY THE EXCLUSIVE REMEDIES SET FORTH IN THIS LIMITED WARRANTY. GNX WILL NOT BE RESPONSIBLE FOR ANY SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGE CAUSED BY ITS PRODUCTS.

GNX PRODUCTS ARE NOT FOR CONSUMER USE. GNX EXPRESSLY DISCLAIMS ALL WARRANTIES ON ANY OF ITS PRODUCTS WHICH MAY BE INCLUDED IN ANY PRODUCT NORMALLY USED FOR PERSONAL OR FAMILY PURPOSES.

Contact GNX by mail at 1337 West 37th Place, Chicago, IL 60609; or phone at (312) 927-5510; if your product is defective, to arrange for its repair or replacement under this warranty.

Repair charges for GIMIX products after warranty period will be \$35.00 per hour per board (minimum \$35.00) plus parts. Customer pays freight charges both ways. If GIMIX determines that replacement is desirable instead, we will notify you. Charges for checking out complete system will be \$500.00 plus parts, freight, and necessary board repairs.

GIMIX, Inc. reserves the right to change pricing, terms, and products specifications at any time without further notice.

**GIMIX** Inc. 1337 WEST 37th PLACE • CHICAGO, ILLINOIS 60609  
(312) 927-5510 • TWX 910-221-4055

*A Member of the CPI Family*

# 68 Micro Journal

10 Years of Dedication to Motorola Users

## Editorial Staff

**Publisher:**

Don Williams Sr.

**Executive Editor:**

Larry Williams

**Production Manager:**

Tom Williams

**Administration:**

*Office Manager:*

Mary Robertson

*Subscriptions:*

Joyce Williams

### Contributing & Associate Editors:

Ron Anderson

Dr. E.M. Bud Pass

Ron Voigts

Art Weller

Doug Lurie

Dr. Theo Elbert

David Lewis

& hundreds more of us

## Contents

Software User Notes	8	Anderson
Basically OS-9	16	Voigts
"C" User Notes	19	Pass
68000 Disk Repair	25	DMW
B1REE	27	Voigts
TEC MCPM-1	29	DMW
Drives Not Ready	37	Mills
P.DATE.FTH	40	Lurie
Classifieds	52	

"Contribute Nothing - Expect Nothing" DMW/1986

# COMPUTER PUBLISHING, INC.

*"Over a Decade of Service"*

"World Wide"

## 68 MICRO JOURNAL CPI

Computer Publishing Center  
5900 Cassandra Smith Road  
PO Box 849  
Hixson, TN 37343

Phone (615) 842-4600 - Telex 510 600-6630

Copyrighted © 1986 by Computer Publishing, Inc.

68 Micro Journal is published 12 times a year by Computer Publishing, Inc. Second Class Postage paid ISSN 0194-5025 at Hixson, TN and additional entries. Postmaster: send form 3597 to 68 Micro Journal, POB 849, Hixson, TN 37343.

## Subscription Rates

1 Year: \$24.50 USA, Canada & Mexico \$34.00 a year.  
Others add \$12.00 a year surface, airmail add \$48.00 a  
year, USA funds! 2 Years \$42.50, 3 Years \$64.50  
plus additional postage, for each additional year.

## Items or Articles for Publication

Articles submitted for publication must include authors name, address, telephone number and date, as well as a statement that the material is original and the property of the submitting author. Articles submitted should be on diskette, Macintosh, OS-9 or FLEX format. All printed items should be dark type and satisfactory for photo-reproduction. No blue ink! No hand written articles - please.

Please do not format with spaces any text indents, chart items, etc. (source listings o.k.) WE will edit in ALL formatting. Text should be flush left column and use ONLY a carriage return to separate paragraphs or other article text items! MacWrite, FLEX TSC, Stylo formatting acceptable.

## Letters & Advertising Copy

Letters to the Editor should be original copy, signed! Letters of gripe as well as praise are acceptable. We reserve the right to reject any letter to the editor or advertising copy material, for any reason we deem advisable.

Advertising Rates: Commercial please contact 68 Micro Journal advertising department. Classified ads must be non-commercial. Minimum of \$15.50 for first 15 words. Add \$.60 per word after the first 15. All classifieds must be pre-paid. No classifieds accepted by telephone.

## The VME BUS and OS-9:

# Ultimate Software for the Ultimate Bus.

Modularity. Flexibility. High Performance. Future growth. These are probably the prime reasons you chose the VME bus. Why not use the same criteria when selecting your system software? That's why you should take a look at Microware's OS-9/68000 Operating System—it's the perfect match for the VME bus.

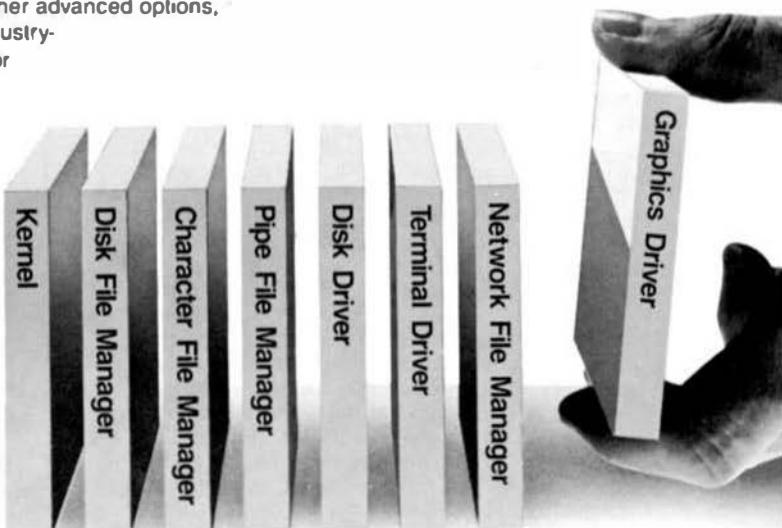
When you're working with VME you must have access to every part of the system. Unlike other operating systems that literally scream KEEP OUT!, OS-9's open architecture invites you to create, adapt, customize and expand. Thanks to its unique modular design, OS-9 naturally fits virtually any system, from simple ROM-based controllers up to large multiuser systems.

And that's just the beginning of the story. OS-9 gives you a complete UNIX-application compatible environment. It is multitasking, real time, and extremely fast. And if you're still not impressed, consider that a complete OS-9 executive and I/O driver package typically fits in less than 24K of RAM or ROM.

Software tools abound for OS-9, including outstanding Microware C, Basic, Fortran, and Pascal compilers. In addition, cross C compilers and cross assemblers are available for VAX systems under Unix or VMS. You can also plug in other advanced options, such as the GSS-DRIVERS™ Virtual Device Interface for industry-standard graphics support, or the OS-9 Network File Manager for high level, hardware-independent networking.

Designed for the most demanding OEM requirements, OS-9's performance and reliability has been proven in an incredible variety of applications. There's nothing like a track record as proof: to date, over 200 OEMs have shipped more than 100,000 OS-9-based systems.

Ask your VME system supplier about OS-9. Or you can install and evaluate OS-9 on your own custom system with a reasonably priced Microware PortPak™. Contact Microware today. We'll send you complete information about OS-9 and a list of quality manufacturers who offer off-the-shelf VME/OS-9 packages.



### MICROWARE.

**Microware Systems Corporation**  
1866 N.W. 114th Street • Des Moines, Iowa 50322  
Phone 515-224-1929 • Telex 910-520-2535

**Microware Japan, Ltd.**  
41-19 Honcho 4-Chome, Funabashi City • Chiba 273  
Japan • Phone 0473 (28) 4493 • Telex 781-299-3122

**Micromaster Scandinavian AB**  
S:t Persgatan 7  
Box 1309  
S75-43 Uppsala  
Sweden  
Phone: 018-138595  
Telex: 76129

**Dr. Rudolf Keil, GmbH**  
Porphystrasse 15  
D-6905 Schriesheim  
West Germany  
Phone: (0 62 03) 67 41  
Telex: 465025

**Eissoft AG**  
Zeilweg 12  
CH-5405 Baden-Dättwil  
Switzerland  
Phone: (056) 83-3377  
Telex: 828275

**Vivaway Ltd.**  
36-38 John Street  
Luton, Bedfordshire, LU1 2JE  
United Kingdom  
Phone: (0582) 423425  
Telex: 825115

**Microprocessor Consultants**  
92 Bynya Road  
Palm Beach 2108  
NSW Australia  
Phone: 02-919-4917

**Microdata Soft**  
97 bis, rue de Colombes  
92400 Courbevoie  
France  
Phone: 1-788-80-80  
Telex: 615405

OS-9 is a trademark of Microware and Motorola. PortPak is a trademark of Microware. GSS-Drivers is a trademark of Graphic Software Systems, Inc. VAX and VMS are trademarks of DEC. Unix is a trademark of AT&T.

# MUSTANG-020 Super SBC <sup>TM</sup>



DATA-COMP proudly presents the first  
Under \$5000 "SUPER MICRO".

## The MUSTANG-020™

### MUSTANG-020™

The MUSTANG-020 68020 SBC provides a powerful, compact, 32 bit computer system featuring the "state of the art" Motorola 68020 "super" micro-processor. It comes standard with 2 megabyte of high-speed SIP dynamic RAM, serial and parallel ports, floppy disk controller, a SASI hard disk interface for intelligent hard disk controllers and a battery backed-up time-of-day clock. Provisions are made for the super powerful Motorola MC68881 floating point math co-processor, for heavy math and number crunching applications. An optional network interface uses one serial (four (4) standard, expandable to 20) as a 125/bit per second network channel. Supports as many as 32 nodes.

The MUSTANG-020 is ideally suited to a wide variety of applications. It provides a cost effective alternative to the other MC68020 systems now available. It is an excellent introductory tool to the world of hi-power, hi-speed new generation "super micros". In practical applications it has numerous applications, ranging from scientific to education. It is already being used by government agencies, labs, universities, business and practically every other critical applications center, worldwide, where true multi-user, multi-tasking needs exist. The MUSTANG-020 is UNIX C level V compatible. Where low cost and power is a must, the MUSTANG-020 is the answer, as many have discovered. Proving that price is not the standard for quality!

As a software development station, a general purpose scientific or small to medium business computer, or a super efficient real-time controller in process control, the MUSTANG-020 is the cost effective choice. With the optional MC68881 floating point math co-processor installed, it has the capability of systems costing many times over its total acquisition cost.

With the DATA-COMP "total package", consisting of a heavy duty metal cabinet, switching power supply with rf/line by-passing, 5 inch DS/DD 80 track floppy, Xebec hard disk controller, 25 megabyte Winchester hard disk, four serial RS-232 ports and a UNIX C level V compatible multi-tasking, multi-user operating system, the price is under \$5000, w/12.5 megahertz system clock (limited time offer). Most all popular high level languages are available at very reasonable cost. The system is expandable to 20 serial ports, at a cost of less than \$65 per port, in multiples of 8 port expansion options.

The system SBC fully populated, quality tested, with 4 serial ports pre-wired and board mounted is available for less than \$3000. Quantity discounts are available for OEM and special applications, in quantity. All that is required to bring to complete "system" standards is a cabinet, power supply, disks and operating system. All these are available as separate items from DATA-COMP.



A special version of the Motorola 020-BUG is installed on each board. 020-BUG is a ROM based debugger package with facilities for downloading and executing user programs from a host system. It includes commands for display and modification of memory, breakpoint capabilities, a powerful assembler/disassembler and numerous system diagnostics. Various 020-BUG system routines, such as I/O handlers are available for user programs.

Normal system speed is 3-4.5 MIPS, with burst up to 10 MIPS, at 16.6 megahertz. Intelligent I/O available for some operating systems.

Hands-on "actual experience sessions", before you buy, are available from DATA-COMP. Call or write for additional information or pricing.

**DATA-COMP**

Installed Systems World-Wide  
OVER 18 YEARS OF DEDICATED QUALITY

**CP** A Division of  
Computer Publishing, Inc.  
5900 Cassandra Smith Road  
Hixson, TN 37343  
Telephone 615 842-4600  
Telex 510 600-6630

# MUSTANG-020

## FEATURES



MUSTANG-020 Benchmarks **		
Time	Seconds	
Type System	12 bit Int. Loop	Register Loop Loop
IBM AT 7300 Xenix Sys 3	9.7	No Registers
AT&T 7300 UNIX PC 68010	7.2	4.3
DEC VAX 11/780 UNIX Berkley 4.2	3.6	3.2
DEC VAX 11/750 " "	5.1	3.2
68008 OS9 68K 8 Mhz	18.0	9.0
68000 " 10 Mhz	6.5	4.0
MUSTANG-020 68020 MC68881 OS9 16 Mhz	2.2	0.68
MUSTANG-020 68020 MC68881 UNIXPLX " "	1.6	1.22

```
** loop: Main()
{
    register long i;
    for (i=0; i < 999999; ++i);
}
```

Estimated MIPS ~ MUSTANG-020 ~ 2.5 MIPS  
Motorola Spec: Burst up to 7 - 8 MIPS - 16 Mhz

### MUSTANG-020™ Software

#### OS-9

OS-9	\$350.00
BasicOS	300.00
C Compiler	400.00
Portos 77	400.00
Microware Pascal	400.00
Overgaard Pascal	900.00
Style-Graph	495.00
Style-Spell	195.00
Style-Merge	175.00
Style-Graph-Spell-Merge	695.00
PAT w/C source	229.00
JUST w/C source	79.95
PAT/JUST Combo	249.50
Sculptor, (see below)	995.00
COM	125.00

#### UniFLEX

UniFLEX	\$450.00
Screen Editor	150.00
Sort-Merge	200.00
BASIC/ProCobol	300.00
C Compiler	350.00
COBOL	790.00
CMODEM w/source	100.00
TMODEM w/source	100.00
X-TALK (see AD)	99.95
Cross Assembler	50.00
Portos 77	490.00
Sculptor, (see below)	995.00

#### Options & Expansions

8 Port expansion RS-232 498.00  
(total of 20 serial ports supported)

Expansion for Motorola I/O Channel  
Modules \$195.00

\*\* All Expansion boards:  
All expansion boards for old style cabinets  
will require the 101 expansion cable.  
Systems ordered with newer PC type  
cabinets do not require this cable.

101 Expansion Cable 539.95

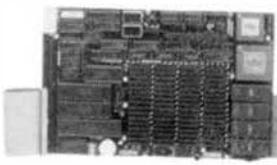
Sculptor: We are USA distributor for  
Sculptor. Call or write for site or multiple  
systems license & discounts. OEM/Dealers.

Special for complete MUSTANG-020™  
systems buyers - Sculptor, \$695.00. Save  
\$300.00!

#### Software Discounts

All MUSTANG-020™ system and board  
buyers are entitled to discounts on all  
listed software: 10-70% depending on item.  
Call or write for quotes. Discounts apply  
after the sale as well.

MUSTANG-020 System component prices - Effective July 1, 1986  
Prices subject to change - call for latest quotes.



MUSTANG-020 (12.50 Mhz)	\$2750.00
** Cabinet (PC or as shown)	\$299.95
5-80 track floppy DS/DD	\$269.95
Floppy cable	\$39.95
OS-9 68K	\$350.00
Winchester cable	\$39.95
Winchester Drive 25 Mbytes	\$895.00
Xebec HD controller	\$395.00
Shipping USA UPS	\$20.00
Total:	\$5059.80

QQ **DISCOUNT LIMITED TIME: Complete System \$1061.00**

**Complete System \$3998.80**

#### OPTIONS ADD:

UniFLEX	\$90.00
MC68881 16 bit math processor	\$275.00
16.67 Mhz MC68020	\$375.00
16.67 Mhz MC68881	\$375.00

**WE WILL NOT BE UNDERSOLD!**

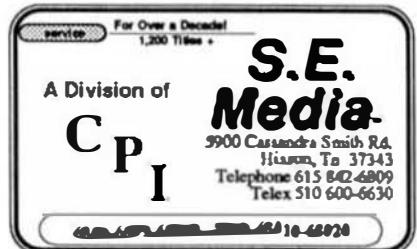
Note: Current OS-9 (Ver. 1.2) does not address the MC68881 - Future  
revisions will. If the 68881 is anticipated in the future, it must be ordered  
with the system, when originally ordered. UniFLEX does support both the  
enhanced code of the 68020 and 68881 now.

**OPTION BOARDS:** \*\* Option boards to be installed in Mustang-020 cabinets  
must be ordered with the extension cable. The cabinet is too tight for  
direct plugin. Or specify our new PC type cabinet, with initial order.

# **PAT - JUST**

**PAT**  
With 'C' Source  
**\$229.00**

**All OS-9  
68XXX  
Systems**



**PAT FROM S. E. MEDIA -- A FULL FEATURED SCREEN ORIENTED TEXT EDITOR** with all the best of PIE. For those who swore by and loved PIE, this is for YOU! All PIE features & much more! Too many features to list. And if you don't like ours, change or add your own. C source included. Easily configured to your CRT terminal, with special configuration section. No sweat!

**68008 - 68000 - 68010 - 68020 OS-9 68K \$229.00**

## **COMBO** ————— **PAT/JUST**

**Special \$249.00**

### **JUST**

**JUST** from S. E. MEDIA -- Text formatter written by Ron Anderson; for dot matrix printers, provides many unique features. Output formatted to the display. User configurable for adapting to other printers. Comes set-up for Epson MX80 with Graflex. Up to 10 imbedded printer control commands. Compensates for double width printing. Includes normal line width, page numbering, margin, indent, paragraph, space, vertical skip lines, page length, centering, fill, justification, etc. Use with PAT or any other text editor. The ONLY stand alone text processor for the 68XXX OS-9 68K, that we have seen. And at a very **LOW PRICE!** Order from: S.E. MEDIA - see catalog this issue.

**68008 - 68000 - 68010 - 68020 OS-9 68K  
With 'C' source \$79.95**



*A Sweetheart of a System & a Sweet Price!*

# MUSTANG-08™

Only From Data-Comp

The Smart Cat Buy!

Now a new addition to the MUSTANG™ series from the DATA-COMP DIVISION of CPI. An economy system, with a *rock bottom price & BIG system features!* The MUSTANG-08™ system will knock the socks off the other 68008 systems now available.

The MUSTANG-08 includes OS9-68K™ and/or Peter Stark's SK\*DOS™. SK\*DOS is a single user, single tasking system that takes up where FLJX™ left off. SK\*DOS is actually a 68XXX FLJX type system (Not a TSC product.)

The OS-9 68K system is a full blown multi-user, multi-tasking 68XXX system. All the popular 68000 OS-9 software runs. It is a speed whiz on disk I/O. Fact is: the MUSTANG-08 is faster on disk access than some other 68XXX systems are on memory cache access. Now, that is fast! And that is just a small part of the story! See benchmark below.

Introductory price of \$1,998.08 (2-80 track DS-DD floppy disk drives). Complete in PC style cabinet, heavy duty switching power supply, if by-passing, ready to run, with your choice of OS-9 68K or SK\*DOS. Add \$750 for a single floppy/25 megabyte hard disk system. For those that waited, DATA-COMP didn't forget.

#### Specifications:

CPU	MC68008	10 MHz
RAM	768K	256K Chips
	No Wait States	
PORTS	2 - RS232	MC68681 DUART
	2 - 8 bit Parallel	MC6821 PIA
CLOCK	MC146818	Real Time Clock
EPROM	16K, 32K or 64K	Selectable
FLOPPY	WD1772	5 1/4 Drives
HARD DISK	Interface Port	WD1002 Board
Size:	5.75 X 8 inches - bolts directly to a floppy or hard disk drive.	

\* Both systems include OS-9 68K or SK\*DOS - Your Choice

## WOW! Benchmark

MUSTANG-08	0 min - 32 sec
Other popular 68008 system	1 min - 05 sec
MUSTANG-020	0 min - 21 sec

C compile time - OS9 68K - Hard Disk all files.  
File: *List* program from K&R, with OS-9 hooks. All systems compiled identical file.



♠ Dual 5" Disk System

**\$1,998.08**

**MC68008**

♠ 25 Megabyte Hard Disk System

**\$2,748.08**

\* Unlike other 68008 systems there are several significant differences. The system is a full 10 Megahertz system. The RAM uses NO wait states, this means full bore MUSTANG type performance.

Also, allowing for addressable ROM/PROM the RAM is the maximum allowed for a 68008. The 68008 can only address a total of 1 Megabytes of RAM. The design allows all the RAM space (for all practical purposes) to be utilized. What is not available to the user is required and reserved for the system.

A RAM disk of 480K can be easily configured, leaving 288K free for program RAM space. The RAM DISK can be configured to any size your application requires (system must have 128K in addition to its other requirements). Leaving the remainder of the original 768K for program use. Sufficient source included (drivers, etc.)

PLEX is a trademark of TSC      MUSTANG-08 is a trademark of CPI  
OS-9 is a trademark of Microware      SK\*DOS is a trademark of StarkKits

## Data-Comp Division



A Decade of Quality Service®  
Systems World-Wide

Computer Publishing, Inc. 5900 Cassandra Smith Road  
Telephone 615 842-4601 - Telex 510 600-6630 Hixson, TN 37343

# SOFTWARE USER NOTES

Ronald W. Anderson  
3540 Sturbridge Court  
Ann Arbor, MI 48105

PANIC

**W**hat an uncomfortable feeling. *MY system is DOWN.* I feel like I have just lost a good friend. My old SUTPC chassis is down for a sudden failure of the GIMIX DMA Disk Controller. I certainly can't complain, having had that system in various forms and sizes for ten years now. This is the first time I have yielded and sent parts off for repair. Last time it was a matter of some bus drivers having failed and I was able to make the repair for myself. This time the system comes up, memory tests OK, but booting yields an immediate error message. I took my circuit boards to work and plugged them into another system. The Processor and memory run fine with another disk controller, and exhibit the identical symptom with my disk controller, so the controller is off to GMX for repair. Meanwhile I can get along with a pair of 5" disk drives and a borrowed Peripheral Technology FD-2 disk controller, provided I remember to get everything I need for a few days onto a few 5" disks at work where I can copy from one disk size to another without too much hassle.

I just mailed a column to '68' MJ, late but not too late, I hope. *Nope Ron, and I certainly appreciate the extra effort you put in getting this to us on time.* - DMW. Things have been hectic all summer, since we decided to put new carpeting in a large portion of our house, which led to several weekends and many evenings of painting and wallpapering. Now Fall is approaching rapidly, and it is time to get at a little clean-up on the PT-69 computer system that goes to school with my daughter Pam. We had considerable trouble with the printer cable last year. I had made it out of ribbon cable, and I learned a lesson. Ribbon cable is excellent when you use the proper ribbon cable connectors. The resulting assembly is very rugged and reliable. However when you use ribbon cable with solder connectors (like the amphenol connector that is the "Standard Centronics" interface connector

for a parallel printer connection, you can expect a wire to break just about every time you connect or disconnect the printer. The reason is the solid conductors of the ribbon cable. It is very hard to strip the insulation from the wire without nicking it, which causes a weak spot that will break after being flexed half a dozen times. This time I "fixed it good". I wired a DB-25 connector on the back of the PT-69 to the parallel port. Then I crimped on a mating DB-25 at one end of the ribbon cable, and the appropriate centronics connector on the other. I wired the DB-25 at the computer so that the wires go "straight through" and no scrambling is required on either end. The Centronics connector is a 36 conductor device but for the Epson and the centronics 737, (which uses an edge connector with the same pin connections) only the first 20 conductors are used, and half of those are ground.

## 68000

The grapevine has it that Peripheral Technology is about to announce its 680xx based computer which will run SK-DOS and OS-9 68K. Maybe now with all the little 680xx single board systems coming around, there will be a flurry of software, as there was for a while after the 6809 systems were introduced. Hopefully, at least, there is a lot of good stuff done in "C" that can be 'ported' to these systems without too much effort. LATER - Yes, it was in the September '68' Micro Journal. It looks like a good way to get started in the 680xx. (added) See DATA-COMP advertising for their MUSTANG-08 this issue.

FOR THOSE WHO NEED TO KNOW

68 MICRO  
JOURNAL™

## Advice?

A number of people have written to me lately for advice. "What should I do with my old 6809 system?" Will there never ever be any more software for these systems? Is it time to bail out and get an IBM compatible and have (as one writer put it) a turnkey system? (Depending on your point of view you might call it a Turkey system). I've discussed this with a few people, and the answers are varied. Some folks are still happy with a 6800 system and what they can do with it. I am presently very happy with my 6809 system running FLEX for what I can do with it (*and what I generally want to do with computing as a hobby and for a living*). Most of those who KNOW the answer to the above questions (for themselves that is), are people like me who are in computing for the fun of it, or for specific applications. If you use your system primarily to learn about operating systems and write utilities, you ought to be happy with FLEX and your 6809 for a long time to come. If you are an equipment collector, (and I use the word in the most positive sense since I have been there myself in the area of photography) and you must have the latest, fastest, and best, you won't be happy with anything less than the latest of the IBM compatibles or a 68020 system. (*added: the IBM types may be the latest, but you don't know what slow really is to you get hung up with one of those, also I don't think best' applies to the IBM types using the 80xx type CPUs. Fact is IBM is dropping market share each month, due to the inefficiency and high asking price of their PC type systems. As to the 68020, well even IBM is going in that direction on their more expensive systems - not PC types. We have had really good hardware for many years. Most of our original 6800 systems will out-perform some of the others's latest. - DMW*)

If you are a computer user who wants to use specialized software for engineering design, such as Finite Element Analysis of structures, PC board layout, or computer aided drafting, you have little choice but to switch to the IBM and clone systems. Such software is available in quantity and at reasonable cost only for those systems.

If you are a business computer user only, and you want to run spreadsheets, accounting packages, databases, and word processors, chances are you already have an IBM compatible with a ton of software (and the probability is 1 in 100 that you read '68' Micro Journal anyway).

Where am I in all this? Well, I have access to some of each at work. I use PI/9, Assembler, "C", Pascal, Extended BASIC, and K-BASIC more or less in that order, regularly on the 6809 system. The Mustang has "C" and Pascal running on it. The 8086/8087 Tandy 1200-HD and the 80286/80287 Tandy 3000 are used to RUN software, not to develop software. We do have Lattice "C" and Turbo Pascal, and of course, GW BASIC (we wondered if the GW stands for "Gee Whiz" as in "Gee whiz, it works"). I've developed a version of JUST in "C" that runs on the Tandy systems in conjunction with PC-Write, the only reasonably usable editor I have found for the PC compatibles, the ravings of some of the users of other packages notwithstanding. I have little or no reason to use the Tandy systems except to run the Engineering Design software that is not available for the 6809 system.

## PATOSTY

A few columns ago I reported having done a utility that would read a STYLO format text file and convert it to a PAT compatible file by inserting CRs now and then within paragraphs, which in Stylo are each one long line. I have done the inverse as well, that is, convert a PAT compatible file to a STYLO compatible file by replacing CRs within paragraphs with spaces. This assumes, of course, that STYLO will be used with the ,JU mode on to justify the text. If that is not the case, CRs should not be deleted anywhere. My main reason for this utility was so that I can write columns using PAT and convert the files easily to STYLO format for use by '68' Micro Journal. It would take a lot fancier program to handle all situations in program conversion, but '68' uses only a few commands in order to insure compatibility between various systems and files, and this one will allow me to do what I need to do.

The fact that I did this utility is in no way a comment on Stylo, which, as I have said before is a very nice editor / text formatter combination. It is more an admission that using two different editors is confusing, and that I am highly prejudiced toward my own effort.

### **Position Independent Code**

I was talking to Don Williams today on the phone and he suggested that I might write a bit about how to write position independent code in assembler. While I can't speak for OS-9 in the 6809 version, since I have not run it, and therefore have never assembled OS-9 code "modules", I can certainly discuss how to write position independent code in the FLEX context, and I think that should be a pretty good start at understanding the process.

First of all, just what is position independent code and why do we sometimes want it? A program is said to be position independent when it is written so that it may be loaded into memory at any valid address, and it will still run. That means in general that the extended addressing mode (and usually the direct addressing mode) are not used in the program. Position independent code in the FLEX environment that uses FLEX routines, however, will have some extended addressing subroutine calls, since the FLEX routines are at fixed locations in memory that do not change when the program is loaded at different addresses. A few examples later will clarify this a little bit.

Position independent code in 6809 assembler is greatly facilitated by the BRA and LBRA instructions, which are both RELATIVE instructions. That is, the assembler calculates the distance from the BRA or LBRA instruction to the routine to be branched TO, and supplies that number to the BRA instruction. If an attempt is made to use a BRA instruction (I include all the conditional branches in this discussion, BNE, BEQ, BMI, etc.) and the distance to the destination of the branch is greater than +127 bytes or -128 bytes, the assembler will give you an error message BRANCH OUT OF RANGE, and you will change the instruction to LBRA, LBNE, LBEQ, etc. Those instructions use a 16 bit offset (distance) and can therefore branch anywhere in 64K of memory. (Arithmetic is done modulo 65536. That is, overflow and underflow are ignored. Try a few examples and you will be convinced that a 16 bit offset can get you anywhere in memory from anywhere else in memory).

Actually, the LBRA and BRA instruction classes take care of half of the battle. The other half is taken care of by the LEA class of instructions and the PCR suffix. LEA stands for Load Effective Address.

When you write a program in position independent code you have to handle variables a little differently than you do with "normal" code. 6800 users or those who have previously used the 6800 or 6502, will be familiar with the process of ORGing an area for variables (frequently on the direct page) and using the assembler RMB instruction to reserve memory bytes for the variables, giving each a label to be used as the variable name. In 6800 code, when you wanted to point the X register at variable COUNT, you simply did a LDX #COUNT, and the X register was loaded with the address of COUNT. In position independent code, however, you don't want an absolute address, but the address relative to where you are right now in the program. The variables are therefore not ORGed at a specific address, but they are RMBed at the beginning or end of the program (or on the stack, but that is another whole subject better left for another time). The assembler knows the address of the label COUNT relative to the program counter value when it encounters the instruction to load X, and you simply use the LEAX COUNT,PCR instruction. X now points at the variable COUNT. The LEAX instruction means that you load X with the ADDRESS of the variable COUNT. LDX COUNT,PCR gets the contents of the variable COUNT into X (assuming COUNT is a 16 bit variable). Similarly LDD COUNT,PCR will get the value of COUNT into ACCD. Note that the LEA.. instructions are limited to the index registers and stack pointers. There is no LEAD instruction, that is.

The point of the whole exercise is that once you have written your program so that you use only relative addressing by using the above set of instructions, the program may be loaded anywhere in memory, and it will still run. Of course you have to jump to the proper "transfer address" after the program is loaded. Earlier I mentioned that FLEX calls such as to PUTCHR or GETCHR must still use extended addressing. PUTCHR EQU \$CD18 appears at the start of your program. LDA #\$0D, JSR PUTCHR as two instructions in sequence, will get a CR sent to the terminal. Of course PUTCHR is located at \$CD18 regardless of where your program is loaded in memory. In general all calls to routines included as part of a position independent program or module must be done via relative addressing modes, and calls to routines outside of the position independent program (i.e. FLEX routine calls) must be done via extended (or absolute) addressing.

Once we have a position independent program, it becomes necessary to have some sort of loader program that can load it anywhere in memory. In the case of FLEX, the user may specify the load address. In the case of OS-9, the operating system determines the load address. The FLEX version is quite simple. FLEX has a LOAD routine that loads a binary file (\$CD30 is the address of this routine.) The System File Control Block at \$C840 must contain the name of a file which has been opened for read as a binary file when this routine is called. FLEX has a Loader Address Offset variable at \$CC1B - \$CC1C, into which can be placed an offset value for the load address. I normally do not use any ORG statement in a position independent program so that it defaults to a load address of \$0000. If I put \$1234 in the Loader Address Offset, and then call LOAD, the program will be loaded starting at address \$1234. Note that if the program has an ORG \$1000, and the Loader Address Offset is set to \$1234, the program will actually be loaded at the specified load address plus the offset or \$2234. This is an unnecessary complication that is eliminated if the program origin is \$0000, to which it defaults if there is no ORG statement. My Position independent programs also have the convention that the starting address is the first byte of the code, so that if the program is loaded at \$1234, a jump to that address will be the correct way to enter the program and run it. Of course the first bytes of the program may well be a branch to another point somewhere else in the code.

With these self imposed rules in mind, I wrote a loader program called LOGO (for LOad and GO with no relation to the language of the same name), and a memory dump program called DUMP. I decided to use the extension .PIC for utilities that are set up for position independent code, and my LOGO utility assumes that extension, and jumps to the load address after the program is loaded. LOGO DUMP 4000 will get DUMP.PIC loaded at \$4000, and then jump to that address to run DUMP. A memory dump program is very much more useful if it can be loaded anywhere in memory, because then, any memory location range can be examined. If you want to dump the Utility Command Space at \$C100, LOGO DUMP 1000 and then examine memory at \$C100. You get the idea. FLEX has a utility SAVE, that is supplied in two versions, SAVE.CMD and SAVE.LOW. The first loads and

executes in the utility command space at \$C100, and the second is used to SAVE utility code from the Utility Command Space, and it loads in low memory (\$0100, I think, but the actual location is incidental). Using PIC, you could rewrite SAVE so that you could LOGO SAVE 8000 and put SAVE smack in the middle of memory, or anywhere else you like.

I am writing this while I am without my Disk Controller and I don't have access to my 8" disks, so a good share of the above is "from memory". Tomorrow at work I can access my 8" disk and copy LOGO and DUMP to the disk that contains this text. I will include the source files for LOGO and DUMP as part of this column since they are good examples of position independent code and a loader for .PIC modules. When I wrote those I also wrote a simple memory move utility that is position independent, but I find that I have very seldom had need for it, so I won't waste the space for its listing here. These utilities use FLEX routines rather extensively, which is the main reason for their shortness.

I have discussed the technique of putting variables on the stack in assembler programs previously, but it has been a very long time, so I will include something on this subject next time. Many 6809 programmers who program in assembler miss the boat, I think, by ignoring the available features of the 6809, primarily the availability of the Y and U registers. The U register specifically is often neglected. U is much easier to use for variables than the System stack register S, since you don't have to put up with remembering the offset of two bytes added for each level of subroutine (for the return address(es)). More on this subject next time.

#### EOF

*Editor's Note: Ron, many are converting to the 68XXX systems. You have for many years - more than anyone else I know of - for most any magazine, given us 'meat & potatoes', as food for thought.*

*What we all need now is some 68XXX basic programming thoughts. Assembler - C - Pascal - BASIC, etc. If you could do it as you did the 6800-6809 era, we would all be mighty grateful. The above discussion is good stuff for old FLEX users (OS-9 types already are into PIC code, as well as many of the 6809 types.) We could soria ride along on your coat-tails and learn as you go along. How 'bout it? Huh?*

DMW

68 MICRO JOURNAL™

FOR THOSE WHO  
NEED TO KNOW

## DUMP

```
5      *
6      *
7      * DUMP.PIC POSITION INDEPENDENT DUMP PROGRAM
8      * USE LOGO UTILITY: LOGO,DUMP,LOADADDRESS
9      *
10     * ENTER WITH P,LOGO,DUMP,LOADADDRESS FOR PRINTED DUMP
11     * PROMPT IS 'COMMAND'. ENTER N(FOR NEXT) AND PAGE NUMBER
12     * FOR EXAMPLE N01 WILL DUMP 0100-01FF
13     * AFTER A PAGE IS DUMPED, ENTER F TO CONTINUE
14     * FORWARD, B TO BACK UP A PAGE, NXX TO GO TO PAGE XX
15     * OR E TO EXIT TO DOS.
16     *
17     *
18     * BY RON ANDERSON
19     * 3540 STURBRIDGE CT.
20     * ANN ARBOR MI 48105
21     * 313 995-1636
22     *
23     *
24          OPT      PAG
```

```
26      * SYSTEM EQUATES
```

```
28      CD15  GETCHR  EQU    $CD15
29      CD18  PUTCHR  EQU    $CD18
30      CD1E  PSTRNG  EQU    $CD1E
31      CC22  SWITCH  EQU    $CC22
32      CD03  WARMS   EQU    $CD03
33      CD3C  OUTHEX  EQU    $CD3C
34      CD24  PCRLF   EQU    SCD24
```

```
36  0000 20  0A      BEGIN  BRA    STARTO
37  0002 01          VERS   FCB    1      FOR VERSION UTILITY
38  0003 43 4F 4D 4D  COMAND FCC    /COMMAND?/
39  000B 04          FCB    4
40  000C C6  FF      STARTO LDB    #$FF
41  000E F7  CC22
42  0011 30  8C EF
43  0014 BD  CD1E
44  0017 5F          START   CLRB   PRINT MESSAGE
45  0018 34  04          PSBB   CLEAR COUNTER
46  001A 8D  7C          BSR    LFCR   SAVE COUNTER
47  001C BD  C015
48  001F 81  46      INCH   JSR    GETCHR  GET COMMAND
49  0021 27  1F      CMPA   #'F   FORWARD TO NEXT MEMORY BLOCK
50  0023 81  45      BEQ    NEWFRM
51  0025 27  0E      CMPA   #'E   EXIT TO DOS
52  0027 81  42      BEQ    EXIT
53  0029 26  0D      CMPA   #'B   GO BACK ONE MEMDRY BLOCK
54          BNE    SKIP1
```

54	002B 6A	8D 006C	DEC	XHI,PCR		
55	002F 6A	8D 0068	DEC	XHI,PCR		
56	0033 20	0D	BRA	NEWFRM		
57	0035 7E	CD03	EXIT	JMP	WARMS	
58	0038 8D	65	SKIP1	BSR	BYTE	GET NEW START ADDR. HI ORDER
59	003A A7	8D 005D		STA	XHI,PCR	SAVE IT
60	003E 6F	8D 005A		CLR	XLO,PCR	START AT BEGINNING OF A PAGE
61	0042 7F	CC22	NEWFRM	CLR	SWITCH	
62	0045 8D	51		BSR	LFCR	
63	0047 30	8D 0050	OUTADR	LEAX	XHI,PCR	
64	004B 8D	3C		BSR	OUT4HS	PRINT ADDRESS OF 1ST BYTE ON LINE
65	004D 5F		OUTMEM	CLRB		CLEAR COUNTER
66	004E AE	8D 0049		LOX	XHI,PCR	GET PAGE START
67	0052 AF	8D 0047		STX	XHI1,PCR	SAVE FOR ASCII PRINT
68	0056 8D	36	LOOP1	BSR	OUT2HS	OUTPUT ONE BYTE
69	0058 5C		MEM1	INCB		COUNT BYTES THIS LINE
70	0059 C1	10		CMPB	#16	SETS BYTES PER LINE
71	005B 26	F9		BNE	LOOP1	
72	005D AF	8D 003A	NXT	STX	XHI,PCR	SAVE X FOR NEXT LINE
73	0061 AE	8D 0038	OUTASC	LDX	XHI1,PCR	GET START OF ASCII PRINT
74	0065 5F			CLRB		CLEAR COUNTER
75	0066 A6	80	OUTAS1	LDA	,X+	GET BYTE
76	0068 84	7F		ANOA	#\$7F	MASK OFF HIGH ORD. BIT
77	006A 81	20		CMPA	#\$20	IS IT PRINTABLE?
78	006C 2C	02		BGE	DOIT	
79	006E 86	2E		LDA	#\$2E	NOT PRINTABLE, PRINT PERIOD
80	0070 8D	CD18	UOIT	JSR	PUTCHR	
81	0073 5C			INCB		COUNT BYTE
82	0074 C1	10		CMPB	#16	
83	0076 26	EE		BNE	OUTAS1	GET ANOTHER BYTE
84	0078 35	04	NXT1	PULB		GET LINE COUNT
85	007A 5C			INCB		INCREMENT IT
86	007B C1	10		CMPB	#16	16 LINES YET?
87	007D 26	04		BNE	NXT2	
88	007F 8D	17		BSR	LFCR	NO, GET SET TO PRINT ANOTHER
89	0081 20	89		BRA	START0	
90	0083 34	04	NXT2	PSHB		
91	0085 8D	11	ENDSTR	BSR	LFCR	END OF A LINE
92	0087 20	BE		BRA	OUTADR	START ANOTHER LINE

94 \* SUBROUTINES FOLLOW \*

96	0089 BU	CD3C	OUT4HS	JSR	OUTHEX
97	008C 30	01		LEAX	1,X
98	008E BU	CD3C	OUT2HS	JSR	OUTHEX
99	0091 30	01		LEAX	1,X
100	0093 86	20	OUTS	LDA	#\$20
101	0095 7E	CD18		JMP	PUTCHR
102	0098 7E	CD24	LFCR	JMP	PCRLF

104 \* TEMPORARY STORAGE \*

106	009B		XHI	RMB	1	
107	009C		XLO	RMB	1	
108	009D		XHI1	RMB	2	
110	009F 8D	0F	BYTE	BSR	INHEX	MODIFIED FROM MIKBUG TO WORK
111	00A1 25	0C		BCS	CONBYT	WITH FLEX GETCHR AT \$710F
113	00A3 48			ASLA		

```

113 00A4 48          ASLA
113 00A5 48          ASLA
113 00A6 48          ASLA
114 00A7 1F  89      TFR   A.B
115 00A9 8D  05      BSR   INHEX
116 00AB 34  04 ABE0  ABA
117 00AF 39          DONBYT RTS
118 00B0 BD  CD15    INHEX  JSR   GETCHR
119 00B3 80  30      SUBA  #$30
120 00B5 2B  11      BMI   ERROR
121 00B7 81  09      CMPA  #09
122 00B9 2F  0A      BLE   DONHEX
123 00BB 81  11      CMPA  #$11
124 00BD 2B  09      BMI   ERROR
125 00BF 81  16      CMPA  #$16
126 00C1 2E  05      BGT   ERROR
127 00C3 80  07      SUBA  #7
128 00C5 1C  FE      DONHEX CLC
129 00C7 39          RTS
130 00C8 1A  01      ERROR  SEC
131 00CA 39          RTS

```

133 END BEGIN TRANSFER ADDRESS FOR FLEX

0 ERROR(S) DETECTED

EOF

## LOGO

LOAD FILE AND GO UTIL09

```

5   *
6   *
7   * LOGO.CMD
8   *
9   * THIS UTILITY ALLOWS LOADING A FILE TO AN OFFSET MEMORY
10  * LOCATION FOR USE WITH POSITION INDEPENDENT CODE
11  * TO LOAD A UTILITY IN A PLACE OTHER THAN IN THE
12  * UTILITY ($C100) AREA IF IT CONFLICTS WITH A PROGRAM
13  * TO BE OPERATED ON BY IT. SUCH POSITION INDEPENDENT
14  * PROGRAMS SHOULD BE ASSEMBLED WITH ORG $0 SO THAT THE
15  * OFFSET SPECIFIED IS THE LOAD ADDRESS AND SHOULD HAVE THE
16  * EXTENSION .PIC. ONLY FILES WITH THIS EXTENSION WILL
17  * LOAD AND GO BY USING THIS UTILITY.
18  *
19  * BY CONVENTION, ALL SUCH POSITION INDEPENDENT PROGRAMS
SHOULD
20  * HAVE THE FIRST ADDRESS AS ENTRY POINT. IT MAY BE DIRECT
21  * OR BE A BRANCH OR JUMP TO ANOTHER START LOCATION.
22  *
23  *
24  * BY RON ANDERSON
25  * 3540 STURBRIDGE CT
26  * ANN ARBOR MI 48105
27  * 313 995-1636
28  *
29  *

```

31	* COMMAND FORMAT: LOGO,FILENAME,LOAD					
ADDR,PARAM1,PARAM2,PARAM3						
32	* PARAMS AS REQUIRED BY THE UTILITY FILE.					
34	* SYSTEM EQUATES					
36	CD03	WARMS	EQU	SCD03	WARMSTART FOR DOS	
37	C840	FCB	EQU	SC840	SYSTEM FCB	
38	CD42	GETHEX	EQU	SCD42	GET 1-4 DIGIT HEX NO. IN X	
39	0406	FMS	EQU	\$0406	FILE MANAGEMENT SYSTEM	
40	CC1B	OFFSET	EQU	SCC1B	HEX OFFSET ADDED TO LOAD ADDRESS	
41	CC0B	SYSDRV	EQU	SCC0B	SYSTEM DRIVE NUMBER	
42	C02D	GETFIL	EQU	SCD2D	FLEX GET FILE NAME FROM COMMAND LINE	
43	CD3F	RPTERR	EQU	SC03F	FLEX REPORT ERROR ROUTINE FOR FILE	
ERRORS	44	0403	FMSCLS	EQU	\$0403	FLEX CLOSE ALL FILES ROUTINE FOR
ERROR	45	C030	LOAD	EQU	SCD30	FLEX BINARY FILE LOAD ROUTINE
	46	*				
	47	C100		ORG	SC100	NORMAL UTILITY
	49	C100 20	01	START	BRA	BEGIN
	50	C102 01		VER	FCB	1
	51	C103 8E	C840	BEGIN	LDX	#FCB
	52	C106 B0	CD20		JSR	GETFIL
	53	C109 B6	CC0B		LDAA	SYSDRV
	54	C10C A7	03		STAA	3,X
	55	C10E 8E	C840		LDX	#FCB
	56	C111 B6	50		LDAA	#'P
	57	C113 A7	0C		STAA	12,X
	58	C115 B6	49		LDAA	#'I
	59	C117 A7	00		STAA	13,X
	60	C119 B6	43		LDAA	#'C
	61	C11B A7	0E		STAA	14,X
CODE						EXT .PIC FOR POSITION INDEPENDENT
	62	C11D B6	01		LDAA	#1
	63	C11F A7	84		STAA	0,X
	64	C121 B0	0406		JSR	FMS
	65	C124 26	13		BNE	ERROR
	66	C126 B6	FF		LDAA	#\$FF
	67	C128 A7	88 3B		STAA	59,X
	68	C12B B0	CD42		JSR	GETHEX
	69	C12E BF	CC1B		STX	OFFSET
	70	C131 BD	CD30		JSR	LOAD
	71	C134 BE	CC1B		LDX	OFFSET
	72	C137 6E	84		JMP	0,X
	73	C139 BD	CD3F	ERROR	JSR	RPTERR
	74	C13C BD	0403		JSR	FMSCLS
	75	C13F 7E	CD03		JMP	WARMS
	76				END	START

0 ERROR(S) DETECTED

EOF

FOR THOSE WHO NEED TO KNOW

68 MICRO  
JOURNAL™

# Basically OS-9

TM

Ron Voigts  
2024 Baldwin Court  
Glendale Heights, IL 60139

## RAM ON WITH MORE RAM!

Some years back at the lab, where I work, we were using an LSI-11 computer on a 16 bit bus. The machine was rated at 64 K bytes. It works out to 32 K words of memory, keeping in mind there is 8 bits per byte and two bytes per word. The system required 4 K words for its use. Leaving us with a grand total of 28 K words for computing. Not much! We were able to run using FORTRAN. Data taking was not impossible, but it we could sure use more memory.

The LSI-11 could be upgraded to more memory. It would involve changing a board. A new Micro-P would be needed to address the extra memory. So we sent the main unit back to California for an upgrade. Weeks later it returned. The boards came wrapped in that fancy anti-static cellophane. The invoice boasted 256 K memory, plus a few other changes. It was ready to be tried!

We plugged it in, turned on the power and booted up. Everything looked fine. The familiar prompt came up. We checked memory. We still had only 28 K words of memory available. We quickly made a call to the California. Now it was 10 AM, here in Illinois. That made it 8 AM on the West Coast. The only person we could find to talk to was the janitor. He was friendly enough, but did not know much about computers. An hour and a half later, we had an engineer on the phone.

Our problem was the system. We had the memory, but couldn't use it. Our system was capable of addressing only the 32 K words. We had 126 K words, but could not access it. We were running a "real time" data acquisition. We would have to change to a multi-user/multi-tasking system to use all the memory. But, our data had to be taken in a "real time" environment. A time sharing system would mean the loss of valuable data at some critical moment. (By the way, at some later time, I will tell you about an attempt to take "real time" data on a time sharing system. I believe, there are users still waiting there turn at a terminal. But that is another story.)

I found a talented individual in our computer department who had experience with the real time system and using the available memory. He recommended a program called VM, which was short for virtual memory. As it turned out a more appropriate name would be virtual disk. The device handler took the unused portion of the memory and treated it as another disk drive on the system. What appeared to be a marginal solution to our problem, turned out to be a great aid. We could use the virtual disk as the system disk, so system stuff loaded incredibly faster. We could compile faster. And during data taking, files could be rapidly created and maintained. Later, after data taking, the files could be downloaded to a real disk.

*Last month I very briefly mentioned that I ran off a Ram Disk. This is my virtual disk and it contains 512K RAM. It is from D.P. Johnson. You can check out the advertisement in the 68' MJ. It looks like a disk drive on the system. I get 2,048 sectors of fast virtual disk space. I load all of my commands onto it. I load compilers like C or Pascal. I load my files. And I still have more room! After entering CHD /R/CMDS, I can enter commands and they execute almost instantly! I run the C compiler and can get a program about the size of the ones I run in the column to compile in a minute or two. Pascal compiles just about as fast.*

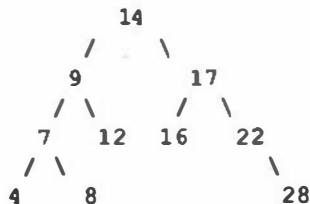
The device descriptor I use is /R and the driver is CCRD. Using SFORMAT supplied with the software, the RAM Disk is formatted as 64 tracks, double sided with 16 sectors per track. That yields 2048 sectors of usable, super fast "disk". D.P. Johnson also includes descriptors to make the RAM disk look like a single or double sided drive, 35 or 40 tracks. This lets BACKUP work, since it looks for identical formatted disks. RAM DISK is made for the color computer. You can check the advertisement, elsewhere in the 68' MJ.

*South East Media's carries OS-9 VDISK for Level 1. This package lets you use the extended memory of a SWTPC or GIMIX CPU card for a virtual disk. I am not totally familiar with it. But, if you are at all interested give them a call. I am sure they would be glad to help.*

## LOOKING THROUGH THE FOREST

I just finished a review for a neat, ISAM like filing system called BTREE, from *Applied Computer Technology*. Their system keeps records in a sequential file. A separate key file is kept. The key file uses a balanced tree structure for keeping track of keys and the pointers to record in the record file. On the average a search through a sequential file takes  $n/2$  comparisons, where  $n$  is the file's record size. For 1000 items, an average of 500 comparisons would be needed. If the keys of the record file were kept in a balanced key structure, the number of comparisons would be  $\log(n)$ . For our 100 item file, we need an average of 3 comparisons! This is considerably much faster and more efficient. Hence, the advantage of a binary key file is speed.

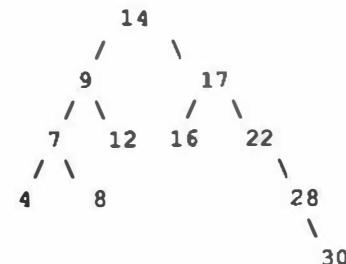
The concept of trees is not a new one. Anyone who has seen a family tree is familiar with the process. There are some basics to know. All trees have some beginning. The first father, if you wish. And to be a father, there must be a son or two. Trees in the computer world are limited to two sons. Usually they are referred to as left and right son, although they could be identified by other designations, like elder and younger, true and false, yes and no and so forth. Across the lineage of different family members, we can talk of cousins, uncles and grandparents. Many times these references become a little contrived. Let's look at a simple example, using a few integers.



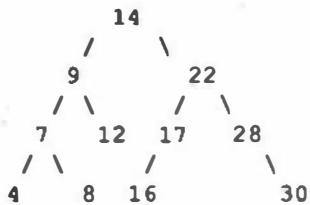
This simple tree consists of 10 integers. The first or main father is 14. It has two sons. The left son is 9 and the right is 17. The number 9 has two sons. Its left son is 7 and the right is 12. If you haven't noticed, all left sons are less than their father and all right sons are greater. The numbers 4, 8, 12, 16, and 28 have no sons. What if the number 25 were added to our family tree? It would become the left son of 28. 25 would be left of 14, 17, and 22. At 28 it would go left. At the end of the lineage, it becomes a left son to 28.

If you don't like the family tree comparison, consider the actual tree. You start with a root. In our case, it is the number 14. It has two branches. They are 9 and 17. 9 has one branch. It is 7. The number 12 is a leaf of 9, since it has no branches. So, any element leading to another is a branch. And elements having no branches are leaves. Collectively, the entire structure could be called a forest.

To get the most efficiency out of the binary tree, it must remain balanced. A balanced tree requires that the length of the left and right branches of an element be no more than  $\pm 1$  in length. Let us say we wanted to add the number 30 to tree. The new structure would appear:



Notice, we are not balanced from 17 on. Its left branch is 1 long while the right is three! This can be corrected with a little pruning. The right side is rotated so that 22 is connected to 14 and 17 is the left branch of 22. The resulting tree would appear:



Now the left and right branches of element 22 are the same size. For elements 17 and 28, the left and right differ by only 1. So our tree is balanced again. Now searching through the tree can be done efficiently again.

## FORESTS OF YOUR OWN

This month's program shows a simple example of putting integers into a tree structure in ascending order. It inputs numbers from the keyboard and places them in the tree structure. A zero terminates the input. Then, the tree is traversed inorder. Every element is visited and printed. The result is the tree's value printed in the order they were stored. The entire program gives a very basic tree type structure and how to store values in ascending order. The program can be modified to include more complex record types. I chose to use the integer, because of its simplicity.

The basics are that a pointer, NDPT, is created to NDTYPE that have some type of information. In our case the information is an integer. But it can be anything you desire. Each record also includes a left and right pointer to the next branch of the tree. Only

three variables are initialized to 'NDPTR'. All other records are strung together with the left and right pointers. It is crucial in a system like this to maintain the integrity of the system.

The function GETND uses Pascal's NEW statement to get a variable of the type NDPTR from Pascal's heap memory. This is a location where dynamic variables can be created. This differs Pascal from other languages. In Basic09 and C, variables are pre-declared. However, Pascal can supply variables referenced by a pointer to some type of variable.

PUTLEFT and PUTRIGHT get the information into a branch of the tree. They make certain that some left or right pointer is pointing to the new branch that was just created. The routine MAKETREE uses GETND to get a new record. It places the actual 'info' into the record and adjust the left and right pointers to NIL.

#### LISTING

```

{ Program: tree
  By: Ron Voigts
  Date: 20-JUL-86
  Compiler: Miroware Pascal Compiler
  To compile: pascal <tree
  To run: pascal tree
}

This is a little program to look at tree
structures in using integers.  |

program tree(input, output);

type ndptr = ^ndtype;
  ndtype = record
    info: integer;
    left: ndptr;
    right: ndptr;
  end;

var p, q, tree: ndptr;
  number: integer;

{ Get a node from the pascal heap }
function getnd: ndptr;
var p: ndptr;
begin
  new(p);
  getnd:=p;
end; { of function getnd }

{ Create a real node in the tree }
function maketree(x: integer): ndptr;
var p: ndptr;
begin
  p:=getnd;
  p^.info:=x;
  p^.left:=nil;
  p^.right:=nil;
  maketree:=p;
end; { of function maketree }

{ Put a node into a left branch }
procedure putleft(p: ndptr; x: integer);
var q: ndptr;
begin
  q:=maketree(x);
  p^.left:=q;
end; { of procedure putleft }

{ Put a node into the right branch }

```

The main program inputs integers. It searches the tree for the correct location of the new piece of information. It uses PUTLEFT and PUTRIGHT to insert newly acquired information into the tree structure. An inorder traversal is then done to demonstrate the tree's structure. The routine INORDER always travels left. When it can't go any more, it backs out, printing what it can. It goes right, until a left traversal is again possible. This continues until the entire tree is traversed. The result is to cover the entire tree structure.

This program should give a fairly good idea how tree structures work. Trees can be created in Basic09 and C language as well. As long as you can create the proper data structures, you can create tree structured files.

That's it for now. Until next time, have a good month!

```

procedure putright(p: ndptr; x: integer);
var q: ndptr;
begin
  q:=maketree(x);
  p^.right:=q;
end; { of procedure putright }

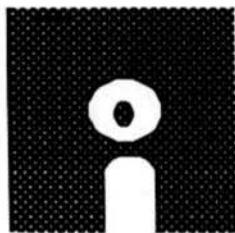
{ Traverse the tree inorder }
procedure inorder(p: ndptr);
begin
  if p<>nil then
    with p^ do
      begin
        inorder(left);
        writeln(info);
        inorder(right)
      end
  end;

begin { of main }
  read(number);
  tree:=maketree(number);
  while (number<>0) do
    begin
      read(number);
      q:=tree;
      p:=tree;
      while ((number>p^.info) and (q<>nil)) do
        begin
          p:=q;
          if number<p^.info then
            q:=p^.left
          else
            q:=p^.right
        end;
        if number=p^.info then
          writeln(number, ' is a duplicate')
        else
          if number>p^.info then
            putleft(p, number)
          else
            putright(p, number)
        end;
      writeln('This is an inorder traversal. ');
      inorder(tree);
    end; { of main }

```

+++

# C User Notes™



E. M. (Bud) Pass, Ph.D.  
Computer Systems Consultants, Inc.  
1454 Latta Lane, N. W.  
Conyers, GA 30207  
404-483-4570/1717

## INTRODUCTION

This chapter continues the discussion of the proposed ANSI C standard and the discussion of common problem areas in the use of the C language and its libraries.

### PROPOSED ANSI C STANDARD

Prototyping is a feature of the proposed standard found in very few current implementations. Its use is optional, but its advantages are so clear that it may be made mandatory in future standards.

Prototyping is the declaration of a function, complete with the number and type of the function's parameters. For example, the library function fwrite is defined as follows:

```
int fwrite(s, l, n, fd)
char *s;
int l, n;
FILE *fd;
{
:
:
}
```

Although it is normally unnecessary, fwrite could be declared as follows:

```
int fwrite();
```

However, with prototyping, fwrite could be pre-declared as follows:

```
int fwrite(char *, int, int, FILE *);
```

which provides the number and type of the parameters of fwrite.

This has the obvious advantage of automatically giving better documentation of the expected function call format to the person writing or maintaining the program. It also has the advantage of providing long-needed information to

conforming compilers to allow them to flag function calls with an incorrect number of parameters or incompatible parameter types or to automatically coerce parameter types in the function call to those expected by the function definition.

For example, the following statement containing fwrite:

```
int fwrite(char *, int, int, FILE *);
char c;
FILE *file;
:
:
if (!fwrite(string, file))
{
:
:
}
```

contains a logical error in that it has two, not four parameters. Currently, most C compilers do not flag this situation, although the lint program can catch it.

For another example, the following statement containing fwrite:

```
int fwrite(char *, int, int, FILE *);
char c;
FILE *file;
:
:
```

*FOR THOSE WHO NEED TO KNOW*

68 MICRO  
JOURNAL™

```
if (!fwrite(c, 1, 1, file))
{
:
:
}
```

contains a logical error in that the first parameter of fwrite must be a pointer to type char, not of type char itself. Again, most current C compilers cannot detect this error.

For still another example, the following statement containing fwrite:

```
int fwrite(char *, int, int, FILE *);
char c[256];
long n = 255L;
FILE *file;
:
:
if (!fwrite(c, 1, n, file))
{
:
:
}
```

contains a logical error (if prototyping is not used) in that the third parameter of fwrite must be of type int. If prototyping is used, the third parameter is automatically coerced to type int from type long. Most current C compilers will not automatically perform the coercion.

Prototyping provides conforming C compilers with several other means of assisting the programmer in ensuring that functions are coded properly. The prototype must match the function header in number and type of arguments exactly. Function parameters may not be redefined in the body of the function.

For example, the following function header for fwrite:

```
int fwrite(char *, int, int, FILE *);
:
:
int fwrite(s, n, l, file)
char s;
int n, l;
FILE *file;
{
    long l;
    :
    :
}
```

would be incorrect since the first parameter should be declared as type pointer to char, not of type char. The function body declaration is also incorrect since it conflicts with the definition in the function header. Some compilers may disallow any redeclaration, not only a conflicting redefinition, of a function parameter.

Prototyping also provides an alternate means of specifying function headers, using similar format and syntax to the prototype.

For example, the function header for fwrite could be coded as follows:

```
int fwrite(char *s, int n,
          int l, FILE *file)
{
:
:
}
```

Trigraphs are specified in the proposed standard to extend the ASCII character set in cases in which the character set is restricted by hardware or other requirements. Examples of such restriction are some implementations of Videotext and in European versions of the ASCII character set in which local characters replace characters required by the C language syntax.

The currently-defined trigraphs are as follows:

```
??= -> #
??< -> (
??> -> )
??( -> [
??) -> ]
??/ -> \
??' -> ^
??! -> !
??- -> ~
```

Trigraphs are recognized at compile time by conforming C compilers. They are not recognized at run time in input data nor at compile time in quoted string literals.

Alphanumeric, octal, and hexadecimal escape codes, all introduced with a reverse slash, may be used to enter these characters and any others in some input data and in all quoted string literals. The proposed standard introduces several new methods of entering such escape codes.

Whereas only octal escape codes were allowed in quoted strings by K&R, the proposed standard also allows hexadecimal escape codes. Octal escape codes are of the form `\ddd`, where `ddd` represents a valid sequence of one to three octal digits (0-7). The first digit after `\` need not be a zero. Hexadecimal escape codes are of the form `\xddd`, where `ddd` represents a valid sequence of one to three hexadecimal digits (0-9,a-f,A-F). The resulting value must not exceed the maximum value which may be contained in an unsigned char type, which is implementation-dependent.

The list of alphanumeric escape codes in the proposed standard is as follows:

- `\a` produces an alert indication, such as a beep or a screen flash
- `\b` moves cursor or print head one location before current location
- `\f` moves cursor or print head to the start of the next logical page
- `\n` moves cursor or print head to the start of the next logical line
- `\r` moves cursor or print head to the start of the current logical line
- `\t` moves cursor or print head to the next logical horizontal tab stop
- `\v` moves cursor or print head to the next logical vertical tab stop
- `\'` produces one single quote character
- `\"` produces one double quote character
- `\?` produces one question mark character
- `\|` produces one reverse slash

If a reverse slash is followed by a character not in the list above and not introducing an octal or hexadecimal constant, the results are undefined in the proposed standard. Most conforming compilers will probably default to the value of the escaped character, for compatibility with current implementations.

A conforming C compiler must provide a file named `limits.h` containing the numerical and logical limits on the implementation. If the limits on the translation-time implementation vary from the limits on the run-time implementation, the stricter set of limits must be provided. The translation limits are normally used to determine some of the characteristics for a compiler when a program is being ported to or written for it. The integral and floating limits may be used for this purpose, but they also may be used to make conforming programs more portable across conforming implementations.

The translation limits and common values for them are as follows:

#### **STATEMENTS\_NEST 15**

maximum nesting level for conditional and iterative structures

#### **CONDITIONAL\_COMPILES\_NEST 6**

maximum nesting level for conditional compilation

#### **DECL\_TYPE\_MODIFIERS 6**

maximum number of declarators modifying a basic declaration type

#### **PARENTS\_NEST 127**

maximum number of declarators modifying a basic declaration type

#### **INTERNAL\_NAME\_LENGTH 31**

maximum number of significant characters in an internal identifier

#### **EXTERNAL\_NAME\_LENGTH 6**

maximum number of significant characters in an external identifier

#### **CASES\_IN\_EXTERNAL\_NAMES 1**

maximum number of alphabetic cases in an external identifier

#### **INTERNAL\_NAMES 1024**

maximum number of internal identifiers in one source file

#### **EXTERN\_NAMES 511**

maximum number of external identifiers in one source file

#### **MACRO\_NAMES 1024**

maximum number of macro identifiers in one source file

#### **CALL\_PARAMS 31**

maximum number of parameters in one function call

**MACRO\_PARAMS 31**

maximum number of parameters in one macro call

**SOURCE\_LINE\_LENGTH 509**

maximum number of characters in one source line

**INCLUDE\_FILES\_NEST 4**

maximum number of nesting levels for included source files

**SWITCH\_CASES 255**

maximum number of case labels in one switch structure

The integral type limits and common values for them are as follows:

**CHAR\_BIT 8**

maximum number of bits for smallest object (char)

**CHAR\_MAX 127**

maximum value of object of type char

**CHAR\_MIN 0**

minimum value of object of type char

**SCHAR\_MAX +127**

maximum value of object of type signed char

**SCHAR\_MIN -127**

minimum value of object of type signed char

**UCHAR\_MAX 255**

maximum value of object of type unsigned char

**SHRT\_MAX +32767**

maximum value of object of type short

**SHRT\_MIN -32767**

minimum value of object of type short

**USHRT\_MAX 65535**

maximum value of object of type unsigned short

**INT\_MAX +32767**

maximum value of object of type int

**INT\_MIN -32767**

minimum value of object of type int

**UINT\_MAX 65535**

maximum value of object of type unsigned int

**LONG\_MAX +2147483647**

maximum value of object of type long

**LONG\_MIN -2147483647**

minimum value of object of type long

**ULONG\_MAX 4294967295**

maximum value of object of type unsigned long

The floating type characteristics and common values for them are as follows:

**FLT\_RADIX 2**

radix of floating-point exponent representation

**FLT\_ROUNDS 0**

floating-point addition rounds (1) or chops (0)

**FLT\_MAX\_EXPONENT +38**

maximum exponent power of 10 that may be represented

**FLT\_MIN\_EXPONENT -38**

minimum exponent power of 10 that may be represented

**FLT\_DIG 6**

maximum number of decimal digits of floating-point precision

Other restrictions and characteristics of conforming C compilers which are not represented in limits.h include the following:

External identifiers beginning with an underscore and all identifiers beginning with two underscores are reserved for the implementation and should not normally be used within a program.

No order or structure of storage assignment by subsequent calls to the library functions malloc, calloc, and realloc may be assumed.

Rules for writing file names and command lines acceptable to the execution environment are implementation-defined.

Certain macro names may be predefined by the user or by the implementation when the translator is invoked.

Macro names conflicting with library function names have precedence over the library function names in the source file in which they appear.

## C PROBLEM AREAS

Function definitions and calls are one of the weaker points of most current C implementations. Few compilers check the number and type of function parameters between the function header and the function call.

The prototyping of the proposed standard addresses this problem, as was described in the preceding section of this chapter. The lint program available under UNIX and certain other implementations also addresses this problem.

There are a few tricks which may be useful in avoiding this problem on current implementations without lint and without prototyping. For example, fwrite could be pre-declared as follows:

```
int fwrite(* char *, int,  
          int, FILE * *);
```

which documents the number and type of the parameters of fwrite. Although the C compiler will not check the number and type of parameters, this type of documentation will assist the user in debugging and modifying C programs until prototyping arrives.

## C PROBLEM

Following are two short C functions which malfunction, as shown in the previous chapter.

This function seems to work correctly itself, but sometimes causes the system to crash after the function has been called.

```
#include <stdio.h>  
  
char *password;  
  
main()  
{  
    :  
    :  
    if (getpasswd(password))  
    :  
}  
  
int getpasswd(password)  
char *password;  
{  
    printf("Enter password: ");  
    return getline(password, 16);  
}
```

The problem is with the original declaration of password. Since it was declared as a pointer of type char, not as an array of type char, no space was allocated for it. Thus, getline would input up to 16 characters into wherever the pointer was pointing.

This function formats a string for subsequent use, but the string usually contains garbage, rather than the correct contents.

```
main()  
{  
    char *form();  
    :  
    printf("%s\n", form(100));  
    :  
}  
  
char *form(value)  
int value;  
{  
    char string[256];  
    sprintf(string, "The value is %d", value);  
    return string;  
}
```

The char array into which the sprintf function places its value is local in scope to the form function. Thus, although the string is formatted correctly and its address is returned properly, its contents are promptly trashed whenever the form function is exited. The variable string may be made static or global to correct the problem.

Following is this month's example C program; it decodes a dump of a binary file into the original binary file. This might be useful when connecting two dissimilar systems with which no common binary modem protocol is available.

```
#include <stdio.h>
#define outbinary "wb" /* output binary
mode */

main(argc, argv)
int argc;
char *argv[];
{
    char *p, string[256];
    int c, i, j, k;
    FILE *in, *out;

    if (argc < 3)
    {
        puts("usage: undump infile outfile");
        puts(" converts dump files to binary.");
        puts(" input format is as follows:");
        puts(" xxxx xx xx ... xx");
        puts(" where xxxx is address");
        puts(" and xx ... xx is data");
        exit(1);
    }
    if (!(in = fopen(argv[1], "r")))
    {
        puts("Can't open ");
        puts(argv[1]);
        putc('\n', stdout);
        exit(1);
    }
    if (!(out = fopen(argv[2], outbinary)))
    {
        puts("Can't open ");
        puts(argv[2]);
        putc('\n', stdout);
        exit(1);
    }
    while (fgets(string, 256, in))
    {
        for (p = string, j = c = k = 0;
             (j < 4) && (!k); ++p, ++j)
        {
            if ((c = *p) <= 0x20)
```

68 MICRO JOURNAL™

FOR THOSE WHO

NEED TO KNOW!

```
++k;
if ((c >= '0') && (c <= '9'))
    c -= '0';
else
    if ((c >= 'A') && (c <= 'F'))
        c -= 'A' - 10;
else
    if ((c >= 'a') && (c <= 'f'))
        c -= 'a' - 10;
else
    ++k;
}
if (k)
    continue;
for (p = string + 4, j = 0; *p; ++p)
{
    if ((c = *p) <= 0x20)
        continue;
    if ((c >= '0') && (c <= '9'))
        c -= '0';
    else
        if ((c >= 'A') && (c <= 'F'))
            c -= 'A' - 10;
        else
            if ((c >= 'a') && (c <= 'f'))
                c -= 'a' - 10;
            else
                continue;
    if (j)
    {
        putc((j << 4) | c, out);
        j = 0;
    }
    else
        j = c | 0x10;
}
if (j)
    putc(j << 4, out);
fclose(in);
fclose(out);
exit(0);
}
```

# 68000 DISK REPAIR

DMW

**B**ack in the good old days our worst fears, which were well founded, was that something would go wrong and all would be lost.

Usually, Murphy struck often. And when he did, he sure knew how to bust you in the most painful way. It seemed that whatever the problem was, there was no fix! *We sure had fun in those days.*

I can remember leaving my 4K system on for weeks at a time, because there were no disks, no tapes, no, not even paper to back up to. If it bombed, which was quite often, you just punched it all in again, with a hex keyboard. Which was certainly a better deal than punching it in bit by bit as some others were doing. *We had it good, a whole nibble at a time.*

Then came the times of invention. First there was the ASR-33 teletype with a paper tape punch. The best percentage of accuracy I was ever able to get out of those I owned, was about 88%, but that was certainly acceptable in those days. Then SWTPC came out with the AC-30 dual tape system. Until the JPC and PERCOM tape systems came along (months later) that was the best going. Slow at 300 baud but 3 times faster than the paper tape and much more dependable. Over

99% in most cases. *I could load 8K BASIC in about 5 minutes*, if memory serves me right. Now that was a winner! And it checksummed o.k. over 90% of the time. That was progress!

Later we began to see articles about others adapting 9 track disk units and other such monsters to our expanded (now up to 16K) micro systems. About then Shugart dropped some hints about something called a 'floppy disk'. Sounded great, but seemed too far off and too expensive for the likes of us folks.

Of course, in due time we all had floppies and that seemed to be all we would ever need. Right?

Well, not quite. Floppies started to come in all sizes and strengths. Just like laundry soap. Got so that a person couldn't rightly decide what was best. Single sided, single density, double sided, double density, 35, 40 or 80 tracks... who knew, what was coming next. Later the 3.5 inch jobs arrived. More confusion. Then the mess started all over again - Winchester technology...

Now, the problem really wasn't the new technology. The problem was that we were not prepared to properly use and maintain the newer and more powerful devices.

Today the problem still remains, to a degree. Especially with the advent of the super micro 68020 from Motorola. (Boy, those guys seem to always be starting something!). Now if your system goes belly-up, who is at fault? The CPU, memory, ROM, operator or could it be the disk drives?

*Today 68020 systems are the most dependable of any systems I have ever worked with.* The systems using these devices, for the most part, are practically bullet proof. Even the software manufacturers are getting better.

The curve for the 68020 has been far more rapid than it ever was for either the 6800 or the 6809. The software is coming along lightyears better than some of the stuff we had back when. The one dip in the curve has been diagnostic software. Yet, that should have been among the very first.

If your system goes down, and you have one of those with the Motorola 020 Bug, you are in luck, it can do an excellent job on everything but the external stuff (it can even give you some info about external devices, but no repair). Mainly your disk drives. And if your disk drive, or more important, the media is at

fault...what? Well, about time, along comes a new software offering called simply Disk Repair Utility. Or even more simply, as the book says, REPAIR.

If you have ever tried to salvage a blown disk, you know what a despair that can be! Especially if it is a hard disk! *One small bit gone astray and it seems like doomsday. Even more important is the simple fact that on the whole, those of us using the 68020 systems have more at stake with these systems than those of yesteryear.*

Repair is a piece of software that should be available to everyone who has anything of value on disks, and who are running OS-9 68K. If I were to be restricted to just one repair utility, Repair would be my first choice. That is how much we value our disk data.

Repair is simple to use. It is designed to facilitate the repair and file recovery from disks, of all types, that have lost their way. Mainly those that have been accidentally deleted or have a sector or two that cannot be read.

Repair is simple to install and use. The CRT can be any 80X24 display.

Suppose you have a problem with /D0. The procedure would be:

\$ repair /d0

After the banner is printed you are prompted to type h for HELP. Fact is, you can't go anywhere in the program without being routed through HELP. No getting lost here.

HELP has the following options, all are self explanatory:

h - help  
o - display disk organization  
d - display sector  
w - write sector  
r - read sector  
c - change byte in sector  
p - get previous sector  
n - get next sector  
x - exit program  
e - change drive  
b - change entire sector  
\$ - shell command

O reads sector zero and displays the OS-9 organization map.

R reads a sector. The current sector number is displayed as well as a hex and ASCII screen dump, very much as would be displayed from the 'dump' utility of OS-9.

Now say you had a problem with sector \$01, byte \$AF, you would read it into a special buffer, by using the 'r' command, sector \$01. After you get the buffer read then, by using the 'c' command (for change byte) you are prompted for the specific byte. by entering the byte address, in our case \$AF, you would simply enter AF. Next you would enter the new byte, in hex. Now use the 'd' command to display the buffer to make sure your change was properly made. At this time the buffer only is changed, nothing on the disk. To get the change back onto the disk you will use the 'w' to write the buffer out to the disk with the new changes. Simple.

By using the other commands you can step through the disk and repair to your hearts content.

The 'b' command has some need of explanation. This command allows you to write any single hex character to the entire buffer, in one swoop. Suppose you wanted to zero the buffer.

Well, just use the 'b' command and type in zero and it will fill the entire buffer with zeros.

One very important feature is the exchange drive command. By being able to read and write also to another drive. The power is that you can read from one and write to the other. Sometimes this is the only way a disk can be completely salvaged.

The '\$' and 'x' commands should pose no problem to the average user.

All in all, this is a tool that every serious 68020 user should have. It can literally save its price a hundred times over, with just one power-out glitch, especially if your disk was in write mode, when the lights flickered. 'nough said?

For more information, contact:

Specialty  
Electronics, Inc.  
909 N. Cleveland Street  
Enid, OK 73703  
405 233-1632

Price \$79.95

EOF

# BTREE

Ron Voights

Sometime back, when the OS-9 Bulletin board was located a Harper College in Palatine, Illinois, I was paging through the messages. There was one from an OS-9 user, who was looking for an ISAM filing system. ISAM is short for Indexed Sequential Access Method. Alas, I could not help him. Now the bulletin board gone. But the good news is there is an ISAM like system available. Let me tell you about it.

It is BTREE from Applied Computer Technology, Inc. Using Basic09, a programmer can maintain and create "ISAM files". Records can be referenced by an alphanumeric key, instead of their relative position in the file. This set of routines can be used by both Level I and Level II systems. The package includes:

#### **Subordinate Procedures:**

*BTree - File handler  
Btree\_Init - Initializes a key file  
InitFiles - Initializes key file and data file  
NewRec - Allocate space for new data record  
DelRec - Delete a data record  
Lock - Lock entire file  
Unlock - Unlock entire file*

#### **Stand-Alone Programs:**

*ExpandRec - Expand existing record size  
BTREE\_Test - Test file handler  
BTREE\_Print - Print Tree blocks*

#### **Example Programs:**

*BTx1 - Salesman file maintenance  
BTx2 - Customer file maintenance  
BTx3 - Invoice file maintenance  
BTx4 - Customer inquiry*

#### **Data Files:**

*SlsmnRecs - Salesman Data File  
SlsmnKeys - Salesman Key File  
CustRecs - Customer Data File  
CustKeys - Customer Key File  
InvRecs - Invoice Data File  
InvKeys - Invoice Key File*

The Subordinate Procedures are routines that can be called from a Basic09 program to create and maintain the file system. The Stand-Alone Programs can be used to examine, test and change the file's structure. The last two categories, Examples and their Data Files, show ways to use the system.

The subordinate Procedures are what you use for your filing system. BTREE is the main one. With it keys can be read or written to the file based on a key. Also, it will return the previous or next key. And it will delete a key. The routine returns a pointer in the file where the record is located. Using Basic09's GET and PUT routine, coupled with SEEK, the records can be read or replaced. Adding or deleting a record uses the procedures NewRec and DelRec. With pointers returned by BTREE, records can be quickly inserted and removed from the data file.

For creating new files there are InitFiles and BTREE\_Init. InitFiles is a procedure that can be called from the main program when creating a new data file and key file. A technique would be to attempt to open a file. If an error #216 (No Such File) is returned, then InitFiles can be employed. Should the data file already exist and another key file is needed, then Btree\_Init can be used. It will initialize new key file.

**FOR THOSE WHO NEED TO KNOW**

**68 MICRO JOURNAL™**

Two other subroutines are supplied. They are Lock and Unlock. These are used to lock and unlock the files you are currently accessing. For Level II users, this prevents the file from being used by someone else on the system. For Level I, these are not necessary. In fact, the lines of code in the source that reference calls to Lock and Unlock can be removed. Alternately, they can be replaced by dummy routines. I created two new assembly language subroutines. The main body of each was:

```
clr b  
rls
```

Then, when Lock and Unlock are called they do nothing, but return to the calling procedure without error. Remember, this is only for Level I users. Also, these are the only subprograms written in assembly language. The rest of the BTree procedures are written in Basic09 code.

The BTree routines come with three stand-alone programs. They are useful for examining, altering and testing the files. ExpandRec is a short procedure that will expand a record size. It inputs the old file and creates a new file with larger records. BTree\_Test permits you to execute the BTree subroutine, specifying the key file, option desired and, of course, the key. And finally, there is BTree\_Print. It dumps the contents of each block in a file. The blocks of the file are printed in sequential order, rather than in tree type structure. This makes it more difficult to follow, but can be useful in looking for a corrupted record.

Last are the example programs and data files. These are very important. They show how to use BTree. The four programs, called BTx1 through BTx4, take you into the various ways to use BTree. BTx1 is a salesman maintenance program. This program uses a number for a key to keep a record on each salesman. BTx2 is the Customer File Maintenance. It uses two key fields and two data files. The salesman files and customer files are used. BTx3, the Invoice File Maintenance program, uses two fields as the key, the invoice number and customer number. Finally, BTx4, the Customer Inquiry, blends the developments of the previous programs and adds processing of multiple records based on a partial key. The net result is you get a number of fine examples of how to use BTree.

The manual explains the workings of all the procedures and programs fairly well. I caution, this material does take careful reading. No real examples occur during the explanation of the procedures. Rather, they are detailed as to what they do. The real learning occurs with the examples. They are all listed in the manual. Looking through them will explain how to use the procedures. All the procedures are documented as to what changes must be made to accommodate your particular application.

As I have previously mentioned, this package comes with the source code. In fact, it is necessary. The record type and key must be changed for your particular file. There are also a couple of numbers that must be changed in the source code. Comment lines in the source are included, indicating where changes should occur. If you are familiar with Basic09, you will find making these changes fairly straight forward.

Tree structures are an efficient way to store data. They speed up hashing procedures. A simple search through a list for a particular key would take  $n/2$  comparisons. 1000 keys in a file would average 500 comparison for a particular key. If the key file is stored in a balanced binary tree, the number of comparisons is  $\log(n)$  or for 1000 items, it takes an average of 3 comparisons. This is definitely an improvement in speed.

If tree structures appeal to you, I suggest looking into BTree from Applied Computer Technology, Inc. You can write your own routines, if you like. But I really suggest looking into this set of Basic09 routines. Everything is put into a neat package with good examples. If you are a Basic09 programmer looking for a good routines to fit your main program, BTree is for you.

**BTREE ROUTINES, Applied Computer Technology Inc., 6435 Summer Ave., Memphis, TN 38134, (901) 377-3503**

+++

# Microcomputer Development System

## TEC MCPM-1

This review will come as a surprise to some of you. Seldom do you ever see something for the IBM PC reviewed in the pages of 68 MICRO JOURNAL! Actually this is the first time that I can remember. However, it is about a CPU that is a member of the 68XX family of CPUs. The 68705XX series.

These particular devices are still much in demand due to their versatility of on-board facilities and ease of programming for fast turn-around. All of us are well aware of the ease of programming throughout the entire 68XX series. But some of you do not realize the other important aspects of these fine CPU devices.

The CMOS versions of these devices are prefixed MC/47XX and are serviced by the MCPM-1 as well as the NMOS series.

The MC687XX series that are programmed by this system are the MC68705P3, P5, U3, U5, R3, R5 and the newer F2 and G2 series. As well as the CMOS types mentioned above.

The main features of these CPU devices, in addition to those most of us are readily familiar with, are various combinations of on-board clock, EPROM, bootstrap ROM, I/O, timers and DA/AD facilities. Somewhere in this series you will probably find one to do about any kind of application desired of an 8 bit micro. And that is the secret of their success and longevity.

What I find most encouraging about hardware and software of this type is that it shores up my contention (sometimes feeling like a *Voice in the Wilderness*, that the 68XX series are a long way from being dead! ) Actually I believe with the increased availability of the CMOS series of devices, and the economy of 8 biters for many, many applications, the 68XX series has good prospects for the future.

Sure, I like the 16 and 32 biters. Use and sell 'em, (this is being written on a 32 biter, but an 8 biter can do it just as well). When it all boils down to the bottom line, the 8 biters still have a lot going. The main problem is that we seem to always *need the very latest, hottest, etc. going*. I wonder why?

When I see hardware and software such as this, I get my spirits boosted. After all 8 biters paid my room & board for many years. I got no complaints!

### HARDWARE

The TEC MCPM-1 Microcomputer Development System is composed of a single board computer, development software and wall mounted plug-in power supply. Driven by an IBM PC/XT/AT, running MS-DOS Version 2.1 or higher.

And you guessed it, the CPU is a 68705P3. With its own RAM, ROM or EPROM and I/O and clocking facilities, it is a complete SBC. Limited by design to programming other 68705XX series. Presently driven only by the IBM PC, in REMOTE mode.

Nothing fancy here, just a solid 8 biter doing its job.

The board measures 7.75 by 3.9 inches and has two Textool ZIF (those zero insertion kind with the release/lock lever, that make insertion and removal of those EPROMS a snap) programming sockets, one 28 pins the other 40 pins, one slide type mode control switch, 2 push button control switches. Provisions for 8K of RAM or EPROM (RAM when in REMOTE mode and EPROM when used in the LOCAL mode). Also one DB25 type RS232 port and five status lights. Standard IBM Modem type cables work real fine.

Power is supplied by a wall mounted plug-in module that delivers 12 vac to the system. That being the solitary power requirement. And the system is completely contained, not requiring a separate EPROM programmer during any stage of the operation.

Also there is a RESET switch for those of us who always feel safer when it is near by.

Operation is quite simple. When the proper code is loaded, pressing the GO button gets the whole operation moving and then you just sit there and wait for the READY lamp to come on. Job finished. In this mode no external or host micro need be connected to the system.

When in the REMOTE mode the system expects to receive commands from a host computer. Indicated by the REMOTE light being lit. In the LOCAL mode the system can be used as a production programmer.

Another of the lamps is a VERIFY lamp. When it is lit it signifies that one of several events has occurred. One, a successful completion of a programming operation. Otherwise, it indicates that a successful self test was completed.

### SOFTWARE

First off, I am not going into much detail of the IBM PC end of this system, mainly because most of you do not have one and for those that do, and would be interested in using this system, you would gain little, as you probably know more about that particular computer than I do. And the thrust of this review is of the TEC MCPM-1 68705XX programmer and the supporting software.



## SPECIAL

### K-BASIC

**K-BASIC under OS-9 and FLEX will compile TSC BASIC, XBASIC and XPC Source Code Files.**



**K-BASIC now makes the multitude of TSC XBASIC Software available for use under OS-9. Transfer your favorite BASIC Programs to OS-9, compile them, Assemble them, and BINGO -- useable, multi-precision, familiar Software is running under favorite Operating System!**

**!!! SPECIAL ~~\$199.00~~ \$99.00 !!!**

# SCULPTOR

### THE PACKAGE

**THE SCULPTOR SYSTEM**  
Sculptor combines a powerful fourth generation language with an efficient database management system. Programmers currently using traditional languages such as Basic and Cobol will be amazed at what Sculptor does to their productivity. With Sculptor you'll find that what used to take a week can be achieved in just a few hours.

### AN ESTABLISHED LEADER

Sculptor was developed by professionals who needed a software development tool with capabilities that were not available in the software market. It was launched in 1981 and since then, with feedback from an ever-increasing customer base, Sculptor has been refined and enhanced to become one of the most adaptable, fast, and above all reliable systems on the market today.

### SYSTEM INDEPENDENCE

Sculptor is available on many different machines and for most popular operating systems including MS-DOS, Unix, Xenix and VMS. The extensive list of supported hardware ranges from small personal computers, through multi-user micros up to large minis and mainframes. Sculptor is constantly being ported to new systems.

### APPLICATION PORTABILITY

Mobility of software between different environments is one of Sculptor's major advantages. You can develop applications on a stand-alone PC and -- without any alterations to the programs -- run them on a large multi-user system. For software writers, this means that their products can reach a wider marketplace than ever before. It is this system portability, together with high-speed development, that makes Sculptor so appealing to value added resellers, hardware manufacturers and software developers of all kinds.

### SPEED AND EFFICIENCY

Sculptor uses a fast and proven indexing technique which provides instant retrieval of data from even the largest of files. Sculptor's fourth generation language is compiled to a compact intermediate code which executes with impressive speed.

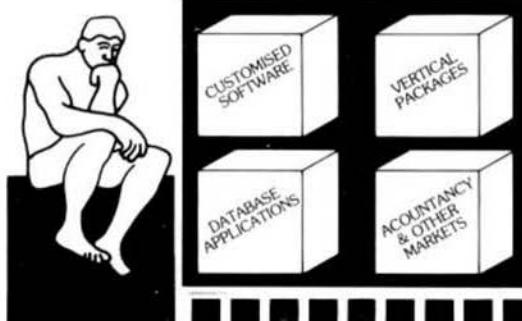
### INTERNATIONALLY ACCEPTED

By using a simple configuration utility, Sculptor can present information in the language and format that you require. This makes it an ideal product for software development almost anywhere in the world. Australia, the Americas and Europe -- Sculptor is already at work in over 20 countries.

### Features

### Features

**Sculptor for 68020  
OS-9 & UniFLEX  
\$995**



**OS-9/UniFLEX**  
IBM PC Zenix .. \$995 / \$199 / \$498  
MS DOS Network .. \* \* \* \*\*\*

**68000 UniFLEX**  
Altos Zenix .. \$1595 / \$319 / \$798  
UNIX .. \* \* \* \*\*\*

**MUSTANG-020™ Users - ask for special discount.**

Sculptor is a Trademark of Microprocessor Developments Ltd.

**!!! Please Specify Your Operating System & Disk Size !!!**

#### Availability Legend--

F = FLEX, CCF = Color Computer FLEX  
O = OS-9, CCO = Color Computer OS-9  
U = UniFLEX  
CCD = Color Computer Disk  
CCT = Color Computer Tape

\* OS-9 is a Trademark of Microware and Motorola  
\* FLEX is a Trademark of Technical Systems Consultants



#### \*\* Shipping \*\*

Add 2% U.S.A.  
(min. \$2.50)  
Add 5% Surface Foreign  
10% Air Foreign



## DISASSEMBLERS

**SUPER SLEUTH** from Computer Systems Consultants  
Interactive Disassembler, extremely **POWERFUL!** Disk File  
Binary/ASCII Examine/Change, Absolute or **FULL**  
Disassembly. XREF Generator, Label "Name Changer", and  
Files of "Standard Label Names" for different Operating  
Systems.

Color Computer SS-50 Bus (all w/ A.L. Source)  
CCD (32K Req'd) Obj. Only \$49.00  
F, \$99.00 - CCF, Obj. Only \$50.00 U, \$100.00  
CCF, w/Source \$99.00 O, \$101.00  
CCO, Obj. Only \$50.00

**DYNAMITE+** -- Excellent standard "Batch Mode" Disassembler.  
Includes XREF Generator and "Standard Label" Files.  
Special OS-9 options w/ OS-9 Version.  
CCF, Obj. Only \$100.00 - CO, Obj. Only \$ 59.95  
F, " " \$100.00 - O, object only \$150.00  
U, " " \$300.00

## PROGRAMMING LANGUAGES

**PL/9** from Windrush Micro Systems -- By Graham Troll. A  
combination Editor Compiler Debugger. Direct source-to-  
object compilation delivering fast, compact, re-entrant,  
ROM-able, PIC, 8 & 16-bit Integers & 6-digit Real  
numbers for all real-world problems. Direct control over  
All System resources, including interrupts. Comprehensive  
library support; simple Machine Code interface; step-by-step  
tracer for instant debugging. 500+ page Manual with  
tutorial guide.

F, CCF - \$198.00

**PASC** from S.E. Media - A Flex9 Compiler with a definite Pascal  
"flavor". Anyone with a bit of Pascal experience should be  
able to begin using PASC to good effect in short order. The  
PASC package comes complete with three sample programs:  
ED (a syntax or structure editor), EDITOR (a simple, public  
domain, screen editor) and CHESS (a simple chess program).  
The PASC package come complete with source (written in  
PASC) and documentation.

FLEX \$95.00

**WHIMSICAL** from S.E. MEDIA Now supports *Real Numbers*.  
"Structured Programming" WITHOUT losing the Speed and  
Control of Assembly Language! Single-pass Compiler  
features unified, user-defined I/O; produces ROMable Code;  
Procedures and Modules (including pre-compiled Modules);  
many "Types" up to 32 bit Integers, 6-digit Real Numbers,  
unlimited sized Arrays (vectors only); Interrupt handling;  
long Variable Names; Variable Initialization; Include  
directive; Conditional compiling; direct Code insertion;  
control of the Stack Pointer; etc. Run-Time subroutines  
inserted as called during compilation. *Normally produces*  
*10% less code than PL/9.*

F and CCF - \$195.00

**KANSAS CITY BASIC** from S.E. Media - Basic for Color  
Computer OS-9 with many new commands and sub-functions  
added. A full implementation of the IF-THEN-ELSE logic is  
included, allowing nesting to 255 levels. Strings are  
supported and a subset of the usual string functions such as  
LEFT\$, RIGHTS\$, MID\$, STRINGS\$, etc. are included.  
Variables are dynamically allocated. Also included are  
additional features such as Peck and Poke. A must for any  
Color Computer user running OS-9.

CoCo OS-9 \$39.95

**C Compiler** from Windrush Micro Systems by James  
McCosh. Full C for FLEX except bit-fields, including an  
Assembler. Requires the TSC Relocating Assembler if user  
desires to implement his own Libraries.

F and CCF - \$295.00

### Availability Legend

F - FLEX, CCF - Color Computer FLEX  
O - OS-9, CCO - Color Computer OS-9  
U - UniFLEX  
CCD - Color Computer Disk  
CCT - Color Computer Tape

\* OS-9 is a Trademark of Microware and Motorola  
\* FLEX is a Trademark of Technical Systems Consultants

!!! Please Specify Your Operating System & Disk Size !!!



Telx 5108006630



5900 Cassandra Smith Rd.  
Hixson, TN 37343

for information  
call (615) 842-4601

**C Compiler** from Introl - Full C except Doubles and Bit  
Fields, streamlined for the 6809. Reliable Compiler. FAST,  
efficient Code. More UNIX Compatible than most.  
FLEX, CCF, OS-9 (Level II ONLY), U - \$575.00

**PASCAL Compiler** from Lueldata -- ISO Based P-Code  
Compiler. Designed especially for Microcomputer Systems.  
Allows linkage to Assembler Code for maximum flexibility.  
F and CCF 5" - \$99.95 F 8" - \$99.95

**PASCAL Compiler** from OmegaSoft (now Certified  
Software) -- For the PROFESSIONAL: ISO Based, Native  
Code Compiler. Primarily for Real-Time and Process  
Control applications. Powerful; Flexible. Requires a  
"Motorola Compatible" Relo. Asmb. and Linking Loader.  
F and CCF - \$425.00 - One Year Maint. \$100.00  
OS-9 68000 Version - \$900.00

**K BASIC** - from S.E. MEDIA -- A "Native Code" BASIC  
Compiler which is now Fully TSC XBASIC compatible.  
The compiler compiles to Assembly Language Source Code.  
A NEW, streamlined, Assembler is now included allowing  
the assembly of LARGE Compiled K-BASIC Programs.  
Conditional assembly reduces Run-time package.  
FLEX, CCF, OS-9 Compiler /Assembler \$199.00

**CRUNCH COBOL** from S.E. MEDIA -- Supports large subset of  
ANSII Level I COBOL with many of the useful Level 2  
features. Full FLEX File Structures, including Random Files  
and the ability to process Keyed Files. Segment and link  
large programs at runtime, or implemented as a set of  
overlays. The System requires 56K and CAN be run with a  
single Disk System. A very popular product.  
FLEX, CCF; Normally \$199.00  
Special Introductory Price \$99.95

**FORTH** from Stearns Electronics -- A CoCo FORTII  
Programming language. Tailored to the CoCo! Supplied on  
Tape, transferable to disk. Written in FAST ML. Many  
CoCo functions (Graphics, Sound, etc.). Includes an Editor,  
Trace, etc. Provides CPU Carry Flag accessibility, Fast  
Task Multiplexing, Clean Interrupt Handling, etc. for the  
"Pro". Excellent "Learning" tool!  
Color Computer ONLY - \$58.95

\*\* Shipping \*\*

Add 2% U.S.A.

(min. \$2.50)

Add 5% Surface Foreign  
10% Air Foreign





## CROSS ASSEMBLERS

TRUE CROSS ASSEMBLERS from Computer Systems Consultants -- Supports 1802/5, Z-80, 6800/1/2/3/8/11/HC11, 6804, 6805/HC05/ 146805, 6809/00/01, 6502 family, 8080/5, 8020/1/2/3/5/C35/39/ 40/48/C48/49/C49/50/8748/49, 8031/51/8751, and 68000 Systems. Assembler and Listing formats same as target CPU's format. Produces machine independent Motorola S-Text.

FLEX, CCF, OS-9, UniFLEX each - \$50.00  
any 3 objects or source (not 68000) - \$100.00  
Set of ALL objects \$200.00 - source \$300.00  
UniFLEX 68000 - \$50.00 - source \$1,000.00

XASM Cross Assemblers for FLEX from S.E. MEDIA -- This set of 6800/1/2/3/5/8, 6301, 6502, 8080/5, and Z80 Cross Assemblers uses the familiar TSC Macro Assembler Command Line and Source Code format, Assembler options, etc., in providing code for the target CPU's.

Complete set, FLEX only - \$150.00

CRASMB from LLOYD I/O -- Supports Motorola's, Intel's, Zilog's, and other's CPU syntax for these 8-Bit microprocessors: 6800, 6801, 6303, 6804, 6805, 6809, 6811 (all varieties); 6502, 1802/5, 8048 family, 8051 family, 8080/85, Z8, 280, and TMS-7000 family. Has MACROS, Local Labels, Label X-REF, Label Length to 30 Chars. Object code formats: Motorola S-Records (text), Intel HEX-Records (text), OS9 (binary), and FLEX (binary). Written in Assembler ... e.g. Very Fast.

Availability: MOTOROLA, INTEL, OTHER, COMPLETE SET

	CPUs	CPUs	CPUs	CPUs
FLEX9	\$150	\$150	\$150	\$399
OS9/6809	\$150	\$150	\$150	\$399
OS9/68K	-----	-----	-----	\$432

CRASMB 16.32 from LLOYD I/O -- Supports Motorola's 68000, and has same features as the 8 bit version. OS9/68K Object code Format allows this cross assembler to be used in developing your programs for OS9/68K on your OS9/6809 computer.

FLEX, CCF, OS-9/6809 \$249.00

## UTILITIES

Basic09 XRef from S.E. Media -- This Basic09 Cross Reference Utility is a Basic09 Program which will produce a "pretty printed" listing with each line numbered, followed by a complete cross referenced listing of all variables, external procedures, and line numbers called. Also includes a Program List Utility which outputs a fast "pretty printed" listing with line numbers. Requires Basic09 or RunB.

O & CCO obj. only - \$39.95; w/ Source - \$79.95

BTREE Routines - Complete set of routines to allow simple implementation of keyed files - for your programs - running under Basic09. A real time saver and should be a part of every serious programmers tool-box.

O & CCO obj. only - \$89.95

Lucidata PASCAL UTILITIES (Requires LUCIDATA Pascal ver 3)

Availability Legend:

F - FLEX, CCF - Color Computer FLEX  
O - OS-9, CCO - Color Computer OS-9  
U - UniFLEX  
CCD - Color Computer Disk  
CCT - Color Computer Tape

\*OS-9 is a Trademark of Microware and Motorola

\*FLEX is a Trademark of Technical Systems Consultants

XREF - produce a Cross Reference Listing of any text; oriented to Pascal Source.

INCLUDE -- Include other Files in a Source Text, including Binary - unlimited nesting.

PROFILER -- provides an Indented, Numbered, "Structogram" of a Pascal Source Text File; view the overall structure of large programs, program integrity, etc. Supplied in Pascal Source Code; requires compilation.

F, CCF -- EACH \$5.00, 8" - \$50.00

DUB from S.E. Media -- A UniFLEX BASIC decompiler Re-Create a Source Listing from UniFLEX Compiled basic Programs. Works w/ ALL Versions of 6809 UniFLEX basic. U - \$219.95

## DATABASE ACCOUNTING

XDMS from Westchester Applied Business Systems - Powerful DBMS; M.L. program will work on a single sided 5" disk, yes is F-A-S-T. XDMS Level I provides an "entry level" System for defining a Data Base, entering and changing the Data, and producing Reports. XDMS Level II adds the POWERFUL "GENERATE" facility with an English Language Command Structure for manipulating the Data to create new file Structures, Sort, Select, Calculate, etc. XDMS Level III adds special "Utilities" which provide additional ease in setting up a Data Base, such as copying old data into new Data Structures, changing System Parameters, etc.

XDMS System Manual - \$24.95

XDMS Lvl I - F & CCF - \$129.95

XDMS Lvl II - F & CCF - \$199.95

XDMS Lvl III - F & CCF - \$269.95

XDMS IV from Westchester Applied Business Systems - XDMS IV is a brand new approach to data management. It not only permits users to describe, enter and retrieve data, but also to process entire files producing customized reports, screen displays and file output. Processing can consist of any of a set of standard high level functions including record and field selection, sorting and aggregation, lookups in other files, special processing of record subsets, custom report formatting, totaling and subtotaling, and presentation of up to three related files as a "database" on user defined output reports.

XDMS IV - F, CCF STAR-DOS, SK\*DOS \$350.00

Upgrades to XDMS IV - \$250.00

## MISCELLANEOUS

TABULA RASA SPREADSHEET from Computer Systems Consultants -- TABULA RASA is similar to DESKTOP/PLAN; provides use of tabular computation schemes used for analysis of business, sales, and economic conditions. Menu-driven; extensive report-generation capabilities. Requires TSC's Extended BASIC.

F and CCF, U - \$50.00, w/ Source - \$100.00

DYNACALC -- Electronic Spread Sheet for the 6809 and 68000.

F, OS-9 and SPECIAL CCF - \$200.00, U - \$395.00

OS-9 68K - \$595.00

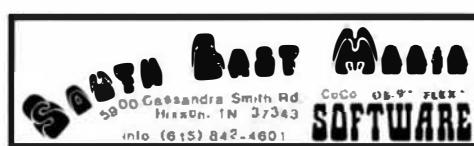
FULL SCREEN INVENTORY/MRP from Computer Systems Consultants -- Use the Full Screen Inventory System/Materials Requirement Planning for maintaining inventories. Keeps item field file in alphabetical order for easier inquiry. Locate and/or print records matching partial or complete item, description, vendor, or attributes; find backorder or below stock levels. Print-outs in item or vendor order. MRP capability for the maintenance and analysis of Hierarchical assemblies of items in the inventory file. Requires TSC's Extended BASIC.

F and CCF, U - \$50.00, w/ Source - \$100.00

\*\* Shipping \*\*

Add 2% U.S.A.  
(min \$2.50)

Add 5% Surface Foreign  
10% Air Foreign



**FULL SCREEN MAILING LIST** from Computer Systems Consultants -- The Full Screen Mailing List System provides a means of maintaining simple mailing lists. Locate all records matching on partial or complete name, city, state, zip, or attributes for Listings or Labels, etc. Requires TSC's Extended BASIC.

F and CCF, U - \$50.00, w/ Source - \$200.00

**DIET-TRAC Forecaster** from S.E. Media -- An XBASIC program that plans a diet in terms of either calories and percentage of carbohydrates, proteins and fats (C P G%) or grams of Carbohydrate. Protein and Fat food exchanges of each of the six basic food groups (vegetable, bread, meat, skim milk, fruit and fat) for a specific individual. Sex, Age, Height, Present Weight, Frame Size, Activity Level and Basal Metabolic Rate for normal individual are taken into account. Ideal weight and sustaining calories for any weight of the above individual are calculated. Provides number of days and daily calendar after weight goal and calorie plan is determined.

F - \$59.95, U - \$89.95

**LOW COST PROGRAM KITS** from Southeast Media -- The following kits are available for FLEX on either 5 or 8 inch disk.

**1. BASIC TOOL-CHEST \$29.95**

BLISTER.CMD: pretty printer

LINEXREF.BAS: line cross-referencer

REMPAC.BAS, SPCPAC.BAS, COMPAC.BAS:

remove superfluous code

STRIP.BAS: superfluous line-numbers stripper

**2. FLEX UTILITIES KIT \$39.95**

CATS.CMD: alphabetically-sorted directory listing

CATD.CMD: date-sorted directory listing

COPYSORT.CMD: file copy, alphabetically

COPYDATE.CMD: file copy, by date-order

FILEDATE.CMD: change file creation date

INFO.CMD (& INFOGMX.CMD): tells disk attributes & contents

RELINK.CMD (& RELINK82): re-orders fragmented free chain

RESQ.CMD: undeletes (recovers) a deleted file

SECTORS.CMD: show sector order in free chain

XL.CMD: super text lister

**3. ASSEMBLERS/DISASSEMBLERS UTILITIES \$39.95**

LINEFEED.CMD: 'modularise' disassembler output

MATH.CMD: decimal, hex, binary, octal conversions & tables

SKIP.CMD: column stripper

**4. WORD - PROCESSOR SUPPORT UTILITIES \$49.95**

FULLSTOP.CMD: checks for capitalization where required

BYTYCIT.BAS (.BAC): Stylo to dot-matrix printer program

NECPRINT.CMD: Stylo to dot-matrix printer filter code

**5. UTILITIES FOR INDEXING \$49.95**

MENU.BAS: selects required program from list below

INDEX.BAC: word index

PIIRASES.BAC: phrase index

CONTENT.BAC: table of contents

INDXSORT.BAC: fast alphabetic sort routine

FORMATER.BAC: produces a 2-column formatted index

APPEND.BAC: append any number of files

CHAR.BIN: line reader

**FULL SCREEN FORMS DISPLAY** from Computer Systems Consultants -- TSC Extended BASIC program supports any Serial Terminal with Cursor Control or Memory-Mapped Video Displays; substantially extends the capabilities of the Program Designer by providing a table-driven method of describing and using Full Screen Displays.

F and CCF, U - \$25.00, w/ Source - \$50.00

!!! Please Specify Your Operating System & Disk Size !!!

**Availability Legends:-**

F - FLEX, CCF - Color Computer FLEX

O - OS-9, CCO - Color Computer OS-9

U - UniFLEX

CD - Color Computer Disk

CCT - Color Computer Tape

\*OS-9 is a Trademark of Microware and Motorola

\*FLEX is a Trademark of Technical Systems Consultants



**SOLVE** from S.E. Media - OS-9 Levels I and II only. A Symbolic Object/Logic Verification & Examine debugger. Including inline debugging, disassemble and assemble. **SOLVE IS THE MOST COMPLETE DEBUGGER** we have seen for the 6809 OS-9 series! **SOLVE** does it all! With a rich selection of monitor, assembler, disassembler, environmental, execution and other miscellaneous commands, **SOLVE** is the **MOST POWERFUL** tool-kit item you can own! Yet, **SOLVE** is simple to use! With complete documentation, a snap! Everyone who has ordered this package has raved! See review - 68 Micro Journal - December 1985. No 'blind' debugging here, full screen displays, rich and complete in information presented. Since review in 68 Micro Journal, this is our fastest mover!

Levels I & II only - OS-9 Regular \$149.95

SPECIAL INTRODUCTION OFFER \$69.95

## DISK UTILITIES

**OS-9 VDisk** from S.E. Media -- For Level I only. Use the Extended Memory capability of your SWTPC or Gimix CPU card (or similar format DAT) for FAST Program Compiles, CMD execution, high speed inter-process communications (without pipe buffers), etc. - **SAVE** that System Memory. Virtual Disk size is variable in 4K increments up to 960K. Same Assembly Required.

Level I OS-9 obj. \$79.95; w/ Source \$149.95

**O-F** from S.E. Media -- Written in BASIC09 (with Source), includes: REFORMAT, a BASIC09 Program that reformats a chosen amount of an OS-9 disk to FLEX Format so it can be used normally by FLEX; and FLEX, a BASIC09 Program that does the actual read or write function to the special O-F Transfer Disk: user-friendly menu driven. Read the FLEX Directory, Delete FLEX Files, Copy both directions, etc. FLEX users use the special disk just like any other FLEX disk

O - 6809/68000 \$79.95

**LSORT** from S.E. Media - A SORT/MERGE package for OS-9 (Level I & II only). Sorts records with fixed lengths or variable lengths. Allows for either ascending or descending sort. Sorting can be done in either ASCII sequence or alternate collating sequence. Right, left or no justification of data fields available. LSORT includes a full set of comments and error messages.

OS-9 \$85.00

**HIER** from S.E. Media - **HIER** is a modern hierachal storage system for users under FLEX. It answers the needs of those who have hard disk capabilities on their systems, or many files on one disk - any size. Using **HIER** a regular (any) FLEX disk (8 - 5 - hard disk) can have sub directories. By this method the problems of assigning unique names to files is less burdensome. Different files with the exact same name may be on the same disk, as long as they are in different directories. For the winchester user this becomes a must. Sub-directories are the modern day



**Shipping**

Add 2% U.S.A.

(min. \$2.50)

Add 5% Surface Foreign

10% Air Foreign





solution that all current large systems use. Each directory looks to FLEX like a regular file, except they have the extension '.DIR'. A full set of directory handling programs are included, making the operation of HIER simple and straightforward. A special install package is included to install HIER to your particular version of FLEX. Some assembly required. Install indicates each byte or reference change needed. Typically - 6 byte changes in source (furnished) and one assembly of HIER is all that is required. No programming required!

\* Introduction Special \* \$69.95

**COPYMULT** from S.E. Media -- Copy LARGE Disks to several smaller disks. FLEX utilities allow the backup of ANY size disk to any SMALLER size diskettes (Hard Disk to floppies, 8" to 5", etc.) by simply inserting diskettes as requested by COPYMULT. No fooling with directory deletions, etc. COPYMULT.CMD understands normal "copy" syntax and keeps up with files copied by maintaining directories for both host and receiving disk system. Also includes BACKUP.CMD to download any size "random" type file; RESTORE.CMD to restructure copied "random" files for copying, or recopying back to the host system; and FREELINK.CMD as a "bonus" utility that "relinks" the free chain of floppy or hard disk, eliminating fragmentation.

Completely documented Assembly Language Source files included.

ALL 4 Programs (FLEX, 8" or 5") \$99.50

**COPYCAT** from Lucidate -- Pascal NOT required. Allows reading TSC Mini-FLEX, SSB DOS68, and Digital Research CP/M Disks while operating under FLEX 1.0, FLEX 2.0, or FLEX 9.0 with 6800 or 6809 Systems. COPYCAT will not perform miracles, but, between the program and the manual, you stand a good chance of accomplishing a transfer. Also includes some Utilities to help out. Programs supplied in Modular Source Code (Assembly Language) to help solve unusual problems.

F and CCF 5" - \$50.00 F 8" - \$65.00

**VIRTUAL TERMINAL** from S.E. Media - Allows one terminal to do the work of several. The user may start as many as eight task on one terminal, under VIRTUAL TERMINAL and switch back and forth between task at will. No need to exit each one; just jump back and forth. Complete with configuration program. The best way to keep up with those background programs.

O & CCO - obj. only - \$49.95

**FLEX DISK UTILITIES** from Computer Systems Consultants - Eight (8) different Assembly Language (w/ Source Code) FLEX Utilities for every FLEX Users Toolbox; Copy a File with CRC Errors; Test Disk for errors; Compare two Disks; a fast Disk Backup Program; Edit Disk Sectors; Linearize Free-Chain on the Disk; print Disk Identification; and Sort and Replace the Disk Directory (in sorted order). -- PLUS -- Ten XBASIC Programs including: A BASIC Resequencer with EXTRAs over "RENUM" like check for missing label definitions, processes Disk to Disk instead of in Memory, etc. Other programs Compare, Merge, or Generate Updates between two BASIC Programs, check BASIC Sequence Numbers, compare two unsequenced files, and 5 Programs for establishing a Master Directory of several Disks, and

Availability Legends -

F - FLEX, CCF - Color Computer FLEX  
 O - OS-9, CCO - Color Computer OS-9  
 U - UniFLEX  
 CDD - Color Computer Disk  
 CCT - Color Computer Tape

\*OS-9 is a Trademark of Microware and Motorola  
 \*FLEX is a Trademark of Technical Systems Consultants

sorting, selecting, updating, and printing paginated listings of these files. A BASIC Cross-Reference Program, written in Assembly Language, which provides an X-Ref Listing of the Variables and Reserved Words in TSC BASIC, XBASIC, and PRECOMPILED BASIC Programs.

ALL Utilities include Source2 (either BASIC or A.L. Source Code).

F and CCF - \$50.00

BASIC Utilities ONLY for UniFLEX -- \$30.00

## COMMUNICATIONS

**CMODEM** Telecommunications Program from Computer Systems Consultants, Inc. -- Menu-Driven; supports Dumb-Terminal Mode, Upload and Download in non-protocol mode, and the CP/M "Modem?" Christensen protocol mode to enable communication capabilities for almost any requirement. Written in "C".

FLEX, CCF, OS-9, UniFLEX; with complete Source \$100.00 without Source \$50.00

UniFLEX 68000 with complete Source \$100.00

**X-TALK** from S.E. Media - X-TALK consists of two disks and a special cable, the hookup enables a 6809 SWTPC computer to dump UniFLEX files directly to the UniFLEX MUSTANG-020. This is the ONLY currently available method to transfer SWTPC 6809 UniFLEX files to a 68000 UniFLEX system. Gimix 6809 users may dump a 6809 UniFLEX file to a 6809 UniFLEX five inch disk and it is readable by the MUSTANG-020. The cable is specially prepared with internal connections to match the non-standard SWTPC S0/9 I/O Db25 connectors. A special SWTPC S+ cable set is also available. Users should specify which SWTPC system he/she wishes to communicate with the MUSTANG-020. The X-TALK software is furnished on two disks. One eight inch disk contains S.E. Media modem program C-MODEM (6809) and the other disk is a MUSTANG-020 five inch disk with C-MODEM (68020). Text and binary files may be directly transferred between the two systems. The C-MODEM programs are unaltered and perform as excellent modern programs also. X-TALK can be purchased with or without the special cables, but this special price is available to registered MUSTANG-020 users only.

X-TALK Complete (cable, 2 disks) \$99.95

X-TALK Software (2 disks only) \$69.95

X-TALK with CMODEM Source \$149.95

**XDATA** from S.E. Media - A COMMUNICATION Package for the UniFLEX Operating System. Use with CP/M, Main Frames, other UniFLEX Systems, etc. Verifies Transmission using checksum or CRC; Re-Transmits bad blocks, etc.

U - \$299.99

## ASSEMBLERS

**ASTRUK09** from S.E. Media -- A "Structured Assembler for the 6809" which requires the TSC Macro Assembler.

F, CCF - \$99.95

**Macro Assembler for TSC** -- The FLEX STANDARD Assembler.

Special -- CCF \$35.00, F \$50.00

**OSM** Extended 6809 Macro Assembler from Lloyd I/O. -- Provides local labels, Motorola S-records, and Intel Hex records; XREF. GeneRate OS-9 Memory modules under FLEX.

FLEX, CCF, OS-9 \$99.00

**Relocating Assembler/Linking Loader** from TSC. -- Use with many of the C and Pascal Compilers.

F, CCF \$150.00

**MACE**, by Graham Trott from Windrush Micro Systems -- Co-Resident Editor and Assembler; fast interactive A.L. Programming for small to medium-sized Programs.

F, CCF - \$75.00

**XMACE** -- MACE w/Cross Assembler for 6800/1/2/3/8 F, CCF - \$98.00

\*\* Shipping \*\*

Add 2% U.S.A.

(min \$2.50)

Add 5% Surface Foreign

10% Air Foreign



## EDITORS & WORD PROCESSING

JUST from S.E. Media -- Text Formatter developed by Ron Anderson; for Dot Matrix Printers, provides many unique features. Output "Formatted" Text to the Display. Use the FPRINT.CMD supplied for producing multiple copies of the "Formatted" Text on the Printer INCLUDING IMBEDDED PRINTER COMMANDS (very useful at other times also, and worth the price of the program by itself). "User Configurable" for adapting to other Printers (comes set up for Epson MX-80 with Grastrax); up to ten (10) imbedded "Printer Control Commands". Compensates for a "Double Width" printed line. Includes the normal line width, margin, indent, paragraph, space, vertical skip lines, page length, page numbering, centering, fill, justification, etc. Use with PAT or any other editor.

\* Now supplied as a two disk set:

Disk #1: JUST2.CMD object file, JUST2.TXT PL9 source: FLEX - CC

Disk #2: JUSTSC object and source in C: FLEX - OS9 - CC

The JTSC and regular JUST C source are two separate programs.

JTSC compiles to a version that expects TSC Word Processor type commands. (pp, sp, oe etc.) Great for your older text files. The C source compiles to a standard syntax JUST.CMD object file. Using JUST syntax (.p, .u, .y etc.) With all JUST functions plus several additional printer formatting functions. Reference the JUSTSC C source. For those wanting an excellent BUDGET PRICED word processor, with features none of the others have. This is it!

Disk (1) - PL9 FLEX only - F & CCF - \$49.95

Disk Set (2) - F & CCF & OS9 (C version) - \$69.95

OS-9 68K000 complete with Source - \$79.95

PAT from S.E. Media - A full feature screen oriented TEXT EDITOR with all the best of "PIE"! For those who swore by and loved only PIE, this is for you! All PIE features and much more! Too many features to list. And if you don't like these, change or add your own. PL-9 source furnished. "C" source available soon. Easily configured to your CRT, with special config section.

Regular FLEX \$129.50

\* SPECIAL INTRODUCTION OFFER \* \$79.95

SPECIAL PAT/JUST COMBO (w/source)

CCF \$99.95

OS-9 68K Version \$229.00

SPECIAL PAT/JUST COMBO 68K \$249.00

Note: JUST in "C" source available for OS-9

CEDRIC from S.E. Media - A screen oriented TEXT EDITOR with availability of 'MENU' aid. Macro definitions, configurable 'permanent definable MACROS' - all standard features and the fastest 'global' functions in the west. A simple, automatic terminal config program makes this a real 'no hassel' product. Only 6K in size, leaving the average system over 165 sectors for text buffer - appx. 14,000 plus of free memory! Extra fine for programming as well as text.

Regular \$129.95

\* SPECIAL INTRODUCTION OFFER FLEX \$69.95

BAS-EDIT from S.E. Media - A TSC BASIC or XBASIC screen editor. Appended to BASIC or XBASIC. BAS-EDIT is transparent to normal BASIC/XBASIC operation. Allows editing while in BASIC/XBASIC. Supports the following functions: OVERLAY, INSERT and DUP LINE. Make editing BASIC/XBASIC programs SIMPLE! A GREAT time and effort saver. Programmers love it! NO more retyping entire lines, etc. Complete with over 25 different CRT terminal configuration overlays.

FLEX, CCF, STAR-DOS Regular \$69.95

Limited Special Offer: \$39.95

### Availability Legend:-

F - FLEX, CCF - Color Computer FLEX

O - OS-9, CCO - Color Computer OS-9

U - UniFLEX

CCD - Color Computer Disk

CCT - Color Computer Tape

\* OS-9 is a trademark of Microware and Motorola

\* FLEX is a trademark of Technical Systems Consultants

!!! Please Specify Your Operating System & Disk Size !!!



### \*\* Shipping \*\*

Add 2% U.S.A.

(min. \$2.50)

Add 5% Surface Foreign  
10% Air Foreign



However, I will say that having looked it over, the screen menus and instructions on the monitor are well presented and should pose no problems to anyone considering this system.

Menu options are:

1. Program 68705 from file
2. Program EPROM from file
3. Create memory image from file
4. Examine memory image
5. Display programmer memory
6. Change Comm port
7. Change fill character
8. Toggle Printer (off/on)
9. Programmer test
0. Exit to DOS

The Comm. port, fill character and printer status are displayed at the bottom of the menu.

Also included on the distribution disk is a well designed tutorial and example session.

In addition to the reference manual for the hardware, as well as a brief but concise overview of the entire system, there are two additional pieces of software that make it a very complete package.

One is a 6805 Simulator Debugger Package, the other is a 6805 Cross Assembler.

The cross assembler takes its input from the host via a text file and creates an output of object code in either the Motorola S-record format or the Intel HEX format, as well as a listing of the file in PC-DOS text format. The default format is the Motorola S-record type.

The assembler is a two pass assembler, with good error checking and reporting.

Output is controlled by 7 software switches. NLF, prevents the generation of a listing. NOF, no object file. NST, no symbol table. When symbols are printed in a table, they are done so in alphabetical order. LTP, list to printer. HEX, change object code to Intel HEX. WOE, wait on error. CMOS, used when assembling to a CMOS target. It traps the STOP and WAIT error opcodes which are unique to CMOS devices.

A full range of pseudo op codes are implemented. DB, define byte. DW, define word. DS, define storage. Also the ORG, END, EQU, INCLUDE, PAGE, PGLEN, TITLE and SBTTL.

Addressing modes are the normal 68XX kinds.

Included with the cross assembler is a 'test' source file used with the tutorial and demonstrates the 6805 instruction set, opcode wise.

The tutorial portion of this package should assist those just beginning to program the 68XX series of devices. It will also be a refresher for some of the older hackers. The documentation is complete enough to get the job done for most any level of experience.

The remaining piece of software is a 6805 Simulator/Debugger program.

The S/D accepts and operates on either Motorola S-record or HEX formats, as well as the symbol table, if available. It outputs a full screen display of the simulated processor's memory, CPU registers, I/O ports, processor timer and interrupt pins.

A 'Softkey' operator is employed on the PC screen. This allows the use of the function keys on the PC keyboard, with the labels for the function keys shown on the bottom of the screen. This saves the user from having to memorize a bunch of extra keyboard function key commands.

The system may be single stepped, run until a specified memory location is reached by the program counter, change the starting address of the screen display, modify memory, registers, I/O ports and flags. Also the simulated clock speed of the processor can be changed to make elapsed time measurements of any series of instructions. Simulated interrupts are supported and the ability to simulate analog inputs to those processors with on-board AD converters.

Again, this package is well supported by error reporting. I have seen, in the past, excellent packages of this sort that did everything but walk the dog. Yet, they fell on their digital noses when it came to giving some sort of useful error reporting. And that is really what it is all about; are there any errors - if so where - what kind, etc? Error reporting is the very heart of a development system.

It has been rightly said that programming is 25% coding and 75% debugging - if you are lucky!

The value of a package such as this is not in any one of its several parts. Individually they are all nice but then there is a lot of 'nice' stuff around. The meat of the thing is that they are not only nice (meaning good, well designed) but that they are a team. Yep, a team, the same as any other group of parts that work together for a common goal. And in that respect, because of the way they work in concert to get the job done in an efficient and speedy manner, they make the overall operation one of the better ones.

This is a *an excellent system* and one I do not hesitate to recommend. Even if they did put it on the IBM PC.

But, on second thought, I reckon it's a good thing. Maybe those folks will come to understand what it was that made us smile all these years!

Sure couldn't do it with a Z80!

The system is priced as follows:

Complete system:	\$495.00
SBC with drivers	349.00
Cross Assembler	100.00
SimDebugger	100.00
Sys Ref Manuals	20.00

Complete system includes all of the above.

Ordering information:

TEC  
PO Box 53  
West Glover, VT 05875  
802 525-3458

+++

# Drives Not Ready

By: J. Gary Mills  
1019 Weatherdon Ave.  
Winnipeg, Manitoba  
Canada P3M 2B5

I don't suppose many FLEX users build their own floppy disk controllers anymore, but I did, and in the process learned a few things about controllers and FLEX disk drivers. One item that came up more than once was the "ready" status of a drive. FLEX checks this status to determine if a disk is present in the drive and the door is closed, and issues the message DRIVES NOT READY if the check fails. According to the FLEX Programmer's Manual, there are two entries in the disk driver jump table that deal with the "ready" check. The Check Drive Ready entry at \$DE0F selects the drive, delays long enough for the motor to come up to speed, and returns with the "ready" status. The Quick Check Drive Ready entry at \$DE12 does the same thing except that the motor start delay is not done. The actual behavior of the disk drivers may vary. The manual states that the TSC minifloppy version always returns a ready status for drives 0 and 1 and a not-ready status for other drives. However, my SWTP FLEX does return a proper status.

Disk controller ICs such as the 1791 that I use have a "ready" input signal which typically is derived from pin 6 of the 34-pin drive cable. The controller IC samples this input before disk read or write operations and sets a bit in its status register if it is not present. The IC will perform seek or restore operations regardless of the state of the ready input. My controller uses the "ready" input, but some others simply tie it true so that FLEX always sees a ready status.

To make matters worse, not all drives produce a "ready" signal on pin 6. There does not seem to be much standardization here. However, my first drives were a pair of BASF 6106 SSDD drives, and they did. The circuit consists of a 300 ms one-shot and a two-bit counter that uses the index pulses to generate a ready signal when a disk is inserted in the drive

and is rotating at the correct speed. When the drive motor shuts off or the door is opened, the signal changes to not-ready after a short delay. The drive and controller combination worked very well, but there was an occasional incident that was annoying. It would occur when a disk access was required just as the drive motor control was timing out. There would be a clank of the head load solenoids, a flash of the drive LED, and FLEX would issue its DRIVES NOT READY message and return to the "+++>" prompt. Sometimes this would happen when I was printing a file on a slow printer so that the motors were cycling on and off. More often, it would be when I pressed <esc> to continue a listing on the screen just as the motor control timed out. It turned out to be due to a design fault in the BASF drives. When the motor control input goes false and quickly back to true, the disk slows below its normal speed, but the "ready" circuit, because of its time delay, momentarily produces a ready indication. This confuses the disk controller IC. The solution to this problem is to reset the drive electronics whenever the motor control input is false. It can be accomplished by connecting two pins on the drive PCB with a short length of wire-wrap wire. This fix should be applicable to BASF 6106 or 6108 drives. Run the wire between pin 2 of test connector TJ2 and pin 1 of jumper connector JJ4. With this simple change, FLEX became very well behaved.

I recently purchased a pair of Quine DSDD half-height drives that were described as being IBM-compatible. They have a lever rather than a door and have no head-load solenoids. I soon discovered that they also have no "ready" output on pin 6. By grounding pin 6 at the controller, I was able to get the drives to work with FLEX, but the operation was not very satisfactory. The best solution would be to add a "ready" circuit to each

FOR THOSE WHO NEED TO KNOW

68 MICRO  
JOURNAL™

drive, but as a temporary solution (which is still in use) I added a tiny board containing an NPN transistor and a 3.0K base resistor. The base signal comes from the internal select line of the drive which may be found on pins 1, 4, 10, and 13 of a 7438. The emitter is grounded, and the collector goes to pin 6 of the 34-pin cable connector. All this circuit does is "turn around" the select signal for the drive, but that is enough to make it behave like a drive with a real "ready" signal.

The following program DISKS.CMD is an example of how the "ready" status can be used. It produces a one-line report on all ready drives like this:

```
Drive: 0 Disk: F9SYS # 65
SYS --- WP 330 sectors free
Drive: 1 Disk: PR-TEXT # 48
SYS WRK -- 97 sectors free
```

I use it in my startup file, and it's also handy whenever I need to find out which disks I have in my drives without opening them and looking at the labels. The program illustrates a couple of FLEX functions which are not widely known, even though they are documented in the FLEX Programmer's Manual. FMS Function 20, Find Next Drive, is given a drive number, which may be -1, and searches up to drive 3 to find the next drive with a ready status. If it doesn't find a ready drive, it returns with an error number in the FCB and the condition code set to NE. Disk driver entry \$DE09, Restore, selects a drive and does a restore operation, which is the same as a seek to track zero. On return, the value \$OB will be in register B if the disk is write-protected. FLEX disk drivers are supposed to adhere to this convention, but it may be that not all of them do. The remainder of the program is fairly conventional, making use of FMS functions to obtain information from the System Information Record of the disk.

I have no information on the various disk controllers available for FLEX, but I presume that all but the simplest of them are able to check the "ready" status of a drive. If any other FLEX user has run into the same kind of problems with "ready", I hope that the information presented here will be of some assistance.

EOF

```
* DISKS command for Flex-9
* One-line report on all mounted disks
*
* J.G.Mills 27 JUN 84
* 1091 Weatherdon Ave
* Winnipeg, Man. R3M 2B5
*
* LIB DISKS.LIB
* SPC 3
* ORG UCA UTILITY COMMAND AREA
DISKS BRA DIINI
* FCB 1 (VERSION)
DIINI LDX #SYSFCB --> FCB
* LDA #$FF DRIVE : -1
* STA FCBDN,X
* DIREP LDA #$XFND REPEAT
* STA FCBFC,X
* JSR FMSCAL . FIND NEXT DRIVE
* BNE DIFIN UNTIL NOT FOUND
* PSHS X
* JSR DDRV+9 . RESTORE DRIVE
* PULS X
* CLR WPBIT
* CMPB #$OB . SET WRITE PROT STATUS
* BNE DIOPS
* COM WPBIT
DIOPS LDA #XOSIR
* STA FCBFC,X
* JSR FMSCAL . OPEN SIR
* BNE DIFIN
* LDA #XGIR
* STA FCBFC,X
* JSR FMSCAL . READ INFO REC
* BNE DIFIN
* BSR PLAY . DISPLAY LINE
* BRA DIREP LOOP
* DIFIN JMP WARMS GOTO FLEX
* SPC 3
* DISPLAY DISK INFORMATION
* X --> FCB
*
PLAY PSHS X
* TFR X,Y
* LDX #MEDRIV
* JSR PSTRNG LABEL
* LDA FCBDN,Y
* ORA #'0
* JSR PUTCHR DRIVE
* LDX #MEDISK
* BSR PDATA LABEL
* BSR PVNAME NAME
* LDX #MENUMB
* BSR PDATA LABEL
* LDB #'1
* LEAX FCBFA,Y
* JSR OUTDEC VOLUME NOM
* LDX #METRES
* BSR PDATA SPACES
* LDX #MESYS
* LDA SYDRV
* BSR PDASN SYSTEM
```

```

LDX    #MEWRK
LDA    WKDRV
BSR    PDASN      WORK
LDX    #MEWP
TST    WPBIT
BNE    PLDWP
LOX    #METWOD
PLDWP BSR    PDATA      WRITE PROTECT
LDB    #1
LEAX   FCBFS,Y
JSR    OUTDEC     FREE SECTORS
IDX    #MEFREE
BSR    PDATA      LABEL
PULS   X,PC
SPC    3
* PRINT DISK ASSIGNMENT
*   X --> "ASSIGNED" MESSAGE
*   Y --> FCB   A = ASSIGNED DRIVE
*
PDASN  TSTA
BMI    PAPRI      IF NOT "ALL"
CMPA   FCBDN,Y   . IF NOT EQUAL
BEQ    PAPRI
LDX    #METRED    . SWITCH TO BLANK MSG
PAPRI  EQU   *     PRINT MESSAGE
SPC    3
* PRINT STRING
*   X --> STRING, EOT
*
PDATA  LDA   ,X+
CMPA   #EOT
BEQ    PDFIN
JSR    PUTCHR
BRA    PDATA
PDFIN  RTS
SPC    3
* PRINT VOLUME NAME
*   Y --> FCB
*
PVNAME LEAX   FCBNAM,Y --> NAME
LDB    #8
PVREP  LDA   ,X+
BNE    PVPRI      REPEAT; GET CHAR
LDA    #SP         . IF ZERO
LDA    #SP         . MAKE IT SPACE
PVPRI  JSR    PUTCHR     . PRINT CHAR
DECB
BNE    PVREP      UNTIL END OF FIELD
RTS
SPC    3
MEDRIV FCC   "Drive: "
FCB    EOT
MEDISK FCC   " Disk: "
FCB    EOT
MENUMB FCC   " 0"
FCB    EOT
METRES FCC   "  "
FCB    EOT
MESYS  FCC   "SYS"
FCB    EOT
MEWRK  FCC   "WRK"
FCB    EOT

```

```

METRED  FCC   "--- "
FCB    EOT
MEWP   FCC   "WP "
FCB    EOT
METWOD FCC   "--- "
FCB    EOT
MEFREE FCC   " sectors free"
FCB    EOT
SPC    3
WPBIT  FCB   0
SPC    3
END    DISKS

```

### EOF

#### \* FLEX Subroutine Linkages

```

FLEX EQU $CD00
WARM$ EQU FLEX+$03 warmstart entry point
PUTCHR EQU FLEX+$18 put character
PSTRNG EQU FLEX+$1E print string with crlf
OUTDEC EQU FLEX+$39 output decimal number

```

#### \* File Management System Entry Points

```

SYSFCB EQU SC840 System FCB
FMSCAL EQU SD406 FMS call

```

#### \* DOS memory map

```

MAP EQU SCC00 start of map
SYDRV EQU MAP+$0B system drive number
WKDRV EQU MAP+$0C work drive number

```

#### \* Structure of an FCB

```

FCBFC EQU 0 function code
FCBDN EQU 3 drive number
FCBNAM EQU 4 file name
FCBFA EQU $0F file attributes
FCBFS EQU $15 file size

```

#### \* Function Codes

```

XGIR EQU 7 get information record
XOSIR EQU 16 open system information record
XFND EQU 20 find next drive

```

#### \* Miscellaneous equates

```

EOT EQU 4 end of text delimiter
SP EQU $20 space
UCA EQU $C100 to $C6FF (Utility Command Area)
DDRV EQU $DE00 to DFFF (Disk Drivers)
OPT LIS

```

### EOF

# p-date.fth

By: R.D. Lurie  
9 Linda St.  
Leominster, MA 01453

## SOME GENERAL COMMENTS ON FORTH

FORTH can be just about anything you want in a computer language. It can be very high level, with everything done for you, so that you hardly need even to know more about your hardware than that it simply exists. Or, FORTH can be so very low level that you might as well write your program in Assembly language. I try to strike a median attitude, possibly biased more toward the high level side of the scale, but I fully intend to use Assembly language definitions whenever appropriate.

Therefore, I hope that all of you who are interested in FORTH will take the time to read what I have to say, and the time to disagree with me whenever you think that I am wrong. I don't want to give the impression that I am any sort of FORTH expert; far from it, I am still a relative beginner with the language. However, I have been writing programs in almost all of the common languages since FORTRAN in 1963, so I have formed some very definite opinions about what I want in a language. As it turns out, I can write programs faster in FORTH than in any other language, including BASIC. The highly structured nature of FORTH probably helps here. However, I am sure that that is only due to my own peculiarities of mind-set, and other people find other languages more to their liking. In any case, now that I am retired, I have more time to spend programming in the language which I prefer. I hope that you will come to like FORTH as much as I do.

There are essentially three versions of FORTH for the 68xx family. There is FIG-FORTH, FORTH-79, and FORTH-83. I have used the FIG-FORTH on both the 6800 and the 6809, and I really have no complaints with either version. I can't say much about FORTH-79, except that I partially converted my 6809 FIG-FORTH to it, and I honestly could not see any advantages. I am now using FORTH-83 as furnished by W. M. Federici. He calls it "FF9", and gives it away! I doubt that you could buy a better version of FORTH for any machine; he offers it for the 6800 and the 6809. The only problem is that there is not enough documentation for the beginner; a beginner will have to buy and absorb several books to be able to use FF9 to its full potential. But don't let that discourage you, since any book on FORTH will teach you enough to get started. Check 68 MICRO JOURNAL, 2/86, for details on getting a copy of FF9.

I cannot comment on the versions of FORTH for the 680xx, except to acknowledge their existence, since I do not have a 680xx machine.

All of my examples will be based on the 6809 and FF9, but I will try to point out any changes which I think might be needed for other versions of FORTH. Also, FLEX is my operating system, but STAR-DOS (SK\*DOS) may work just as well. At the last I heard, FF9 did not work reliably with any version of FLEX for the CoCo, but this may have changed; only Wilson Federici can tell us the latest there.

Incidentally, does anyone know of a FORTH for OS-9?

### .DATE

The application .DATE, as FORTH programmers sometimes call a program, prints the current system date in what I call "the common form", that is, the name of the month is spelled out fully, followed by a comma, the number of the day, a comma, and the year as four digits. As is usual with FORTH, .DATE can stand alone as a useful command to use from the terminal, but it can also be called from definitions in a much larger program.

However, .DATE is also useful in illustrating several programming techniques:

1. CONSTANTS used as RAM pointers.
2. Use of CASE, and
3. Number formatting.

### CONSTANTS

Three CONSTANTS are defined as hex numbers, for convenience. Each CONSTANT is to be used as a pointer to a single byte in system RAM. These three pointers can point to any section of memory where the needed data might be stored; there is no need for them to be consecutive numbers. These just happen to be the three addresses used by 6809 FLEX. Go ahead and change them, if you need to. PRINT THE NAME OF THE MONTH

This definition requires that there be a number on the Data Stack before execution. This is what is meant by the rather cryptic comment: ( n ... ) which shows what happens to the stack during the execution of .MONTH. The "n" represents any 16-bit, signed integer on the top of the stack. The "..." represents the execution of the word, and nothing prior to the delimiter shows that the only change to the stack was the removal of "n". The parentheses are simply comment delimiters. Incidentally, all hell could break loose during the time represented by "...", but, as long as it is transparent to the programmer, nobody cares!

.MONTH makes use of CASE. This form of CASE was first presented by Dr. C. E. Eaker in FORTH DIMENSIONS, II/3. It is based on number matching, so that it is simple, fast, and reliable; it resembles the "case" or "switch" of other languages.

.MONTH is unusual in that it does not need any error trapping. Of course, they would do no harm, by why do it if it's not needed? Test it yourself by executing the following definition: : MM 14 0 DO CR 1 DUP 2 .R 2 SPACES .MONTH LOOP CR ; There is no way I know of in which an out-of-range number can do harm with this form of CASE .

The four words: CASE OF ENDOF ENDCASE make up a required set. The program will crash if they are not all used, and used in the proper order. The selection is initiated by the word CASE . Any legal expression may follow CASE , prior to the use of OF ; however, the top-most 16-bit integer on the Data Stack will be used for comparison to the selector key integer which just precedes OF .

The selector key must appear as the last word in the definition prior to OF, and it must be an integer, a CONSTANT, or an expression which evaluates into an integer. OF DROPS the value on the Data Stack during the comparison, so an extra copy must be made before the execution of OF if the value will be needed elsewhere, even between OF and ENDOF.

The expression between OF and ENDOF is executed if a match is found between the top value of the Data Stack and the selector key. The full expression is executed, no matter how long or how short it is. ENDOF then causes a branch to the expression immediately following ENDCASE .

If no match is found, then execution branches to an optional expression immediately preceding ENDCASE. As in the situation shown here, if there is no such expression, execution proceeds normally following ENDCASE.

I'll discuss CASE in more detail another time. PRINT THE DATE

.DATE begins by using C@ to fetch the byte pointed to by SYSMON . This byte is used as the number on the Data Stack required by .MONTH, which prints the name of the month.

The day, when printed, may consist of either one or two digits, so provision must be made for this condition. The byte returned by C@ from SYSDAY is duplicated and compared to 10. If the byte is less than 10, that is, has a value of 1-9, then two columns are allocated for printing it; however, if the byte has a value of 10 or more, then three columns are allocated. In either case, the first of the allocated columns shows as the empty space between the name of the month and the number of the day.

A short ASCII string is then printed which contains a comma, a space, and "19", which is the century number. This is followed by the number for the year, fetched in the same way as the other parts of the date. Notice that by allocating only two columns to the number of the year, the two digits are up against the "19" of the string, and no spaces intervene. Obviously, this part of the program will need changing after about 13 years, but I will leave that to you to do.

As with any FORTH application, execution is initiated by entering: DATE System Requirements.

For those of you with FF9 and FLEX, I suggest that .DATE be entered just as it is printed here as a FLEX text file named P-DATE.FTH. You can omit the comments, but I recommend that the listing otherwise be typed in just as it is shown here. That will make it easier to find any typing errors, which can be a problem when entering a FORTH listing. Remember that the spaces between words are not significant, just so long as there is at least one space. Remember, also, that the punctuation is a part of the program and cannot be skipped.

If you are using another version of FORTH which uses FLEX DOS, then you should still be able to enter P-DATE.TXT just as shown here. I think that it is valid for FIG-FORTH and FORTH-79.

Those of you who prefer the more conventional FORTH screens can, of course, enter .DATE that way. I suggest that you plan on using 3 screens, so that you do not split a definition across screen boundaries.

EOF

LIST P-DATE LST

```
( ..... )  
( Constants )  
( ..... )
```

100

CC0E CONSTANT SYSDAY  
CC0F CONSTANT SYSDAY  
CC10 CONSTANT SYSYR

REGIMENT

( ..... )  
( Print the name of the month from the supplied number )  
( Nothing is printed if the number is out of range, this is 6 )  
( function of the way CASE has been defined within the FORTH )  
( compiler )  
( ..... )

MARCH (1973)

CASE  
1 OF "January" ENDOF 2 DF "February" ENDOF  
3 OF "March" ENDOF 4 OF "April" ENDOF  
5 OF "May" ENDOF 6 DF "June" ENDOF  
7 OF "July" ENDOF 8 DF "August" ENDOF  
9 OF "September" ENDOF 10 DF "October" ENDOF  
11 OF "November" ENDOF 12 DF "December" ENDOF  
ENDCASE.

( ..... )  
( Print the date in the form Month dd, 19yy )  
( ..... )

DATE (—)  
SYSDM CO JNTH ( print the month )  
SYSDW CO DYP 10 C ( print the day )  
IF 2 R  
ELSE 3 R THEN  
., 19" ( print the century )  
SYSDY CO 2 R, ( print the year )

20

# BIT-BUCKET

By: All of us.....

Dear Don,

Last time I wrote I promised a little discussion on the LSET and RSET statements, but let me begin by saying that the reason they are not normally encountered in 'ordinary' XBASIC programs is probably because they would serve no purpose in programs which scroll continuously. This will become clear in the discussion following:

**LSET** Let's suppose we have a screen-oriented display with an area reserved for error-messages of various kinds, and that a message "Illegal move entry" has just been displayed. Now a new message has to be overlaid in the same area, say "Syntax error". The problem is that if it's simply overlaid we'll see the message "Syntax error entry", so obviously we have to delete the old message in some way. This, of course, leads us into the different methods which could be employed. For example:

a. If the message line extends right to the edge of the screen one could simply position the cursor at the start of the message, issue an "erase-to-end-of-line" code, and write in the new message.

b. If the message area has some vital part of, let's assume, a game-board to the right of it, we can't use (a) above as it would wipe out part of the game-board. In this situation we could position the cursor as above, print maybe 30 SPACES, re-position the cursor, and then write the new message, OR

c. We could use LSET. To do this nicely, we would, by way of a demo program, write:

```
10 A$="           "      (just 20 SPACES)
20 INPUT Q$: LSET A$ = Q$
30 PRINT A$; "X": GOTO 20
```

Run it, and try entering responses of up to 20 characters in response to the prompt. Note that no matter how long or short the response is, LSET will always pad out to 20 characters with SPACES, before displaying the "X" (which is included simply to mark the end of A\$). If the response is over 20 characters long, LSET will truncate it down to 20 characters. There is another little advantage to using LSET, and that is that having printed our message, the cursor will be nicely positioned at our point "X", ready for whatever has to follow next - possibly to request a new and acceptable input this time around.

**RSET** What can I say about this? Except that whereas LSET left-justifies the new message, RSET right-justifies it, padding from the left with SPACES. This would be more useful for displaying 'card' entries, as the amounts would be neatly columnised at the right.

I'm now going to digress, and discuss XBASIC a little more generally. First, another error in the manual, which states that the maximum length of an XBASIC line is 127 characters. Not true! Lines can be up to 255

characters in length. Secondly, an undocumented feature, namely that if one enters LIST -100 in response to the READY prompt, your program will be listed from the beginning up to and including Line 100. The reverse, LIST 100-, does not, however, LIST from Line 100 to the end of the program. This feature does work in RBASIC (see below)!

Thirdly, and much more importantly, Dr. Piasecess, of Umzinto University, Southern Africa, has pointed out to me a very serious flaw in XBASIC's floating-point math operations, which can produce completely erroneous results when dealing with very large, or very small, numbers. I've supplied him with a patch to take care of difficulties with the MULTIPLY operation, but the DIVIDE patch, though not a difficult one in itself, is not so easy to fit in.

Which brings me to a subject which should be of interest to a large number of our readers. As you know, I've been studying the inner workings of XBASIC for some years now, working out patches for this and that, or modifying it so it can call EDIT (an XBASIC line-editor) directly from XBASIC itself. All this has really been stop-gap, finger-in-the-dyke patching, so some time ago I decided to write my own BASIC - to be called RBASIC ('R' for 'Robert', my first name).

It's almost complete now, and though I've obviously had to make it compatible with XBASIC, so you don't have to scrap all your old XBASIC programs, it differs considerably in the following major respects:

1. It's been written directly in optimised 6809 code, instead of 'warmed-over' 6800 code.
2. It doesn't have any of the known XBASIC 'bugs', such as the erroneous 'Missing Parentheses' or flawed floating-point math.
3. 'I' has been added as a short form of 'INPUT', similar to '?' for 'PRINT'.
4. It includes the function 'ARC', so that ARCSIN, ARCCOS and ARCTAN may be implemented, though for compatibility's sake I've had to retain the original ATM function as well.
5. CHAIN allows CHAINing to a file with any extension, though for now the file to which it CHAINS has to be of the same type as that currently in memory, i.e., either all BAC files, or all BAS files.
6. Most importantly, it includes a built-in line-editor with the following features:
  - a. Instant recall of an erroneously-entered line which has been rejected by XBASIC.
  - b. Editing of any specified program line.
  - c. Full cursor-control, LEFT, RIGHT, UP and DOWN. Why UP and DOWN, you ask? Just in case you're editing a 255-character line, and wish to do some editing in the middle of the second displayed line.

- d. Express **RIGHT/LEFT** cursor positioning for moving quickly from one end of a line to the other.
- e. **INSERT**, **DELETE** and **OVERLAY**, plus **SPLIT** and **MELD**. **SPLIT** will (as its name implies) split a line into two individual lines at the cursor-point, usually at a colon. **MELD**, on the other hand, will meld, or join together, two successive program lines, with automatic deletion of the second-line's line-number, and its replacement by a colon. What do you think of that?

Keeping in mind that BEDIT, if it were used at all, took up between 1500 and 2000 bytes, you may be wondering just how long this new RBASIC is, seeing that it includes the best features of BEDIT, plus the extras mentioned above, such as ARC, plus other features too numerous to detail. Would you believe that in spite of all these goodies it's still currently between 1500 and 2000 bytes shorter than XBASIC? In addition, due to improved algorithms, more efficient coding and so on, it should execute faster than XBASIC. Time-comparisons will come later, when we've added as many goodies as possible, before marketing later on this year. So if you've been concerned about some of the limitations of XBASIC, or are considering an 'add-on' line-editor (which only eats up vital memory), you should maybe hold off for a while.

I hope sincerely that you won't take this as a put-down of IBASIC. This has been a wonderful program (one of the best BASICs around, in my opinion), but its days are numbered, especially as TSC no longer seems to be supporting it, and I've given up on just patching it! One can only go so far in converting a prop-driven plane into a jet, then there comes a time when the project has to be scrapped, and the new jet built from the ground up.

I'd like to hear from friends old and new if you have any ideas for incorporating into RBASIC, though I'd still like to keep it as compact as I possibly can. My present thoughts are to create add-on specialty math packages, such as a Statistical package, a Hyperbolic Functions package, and so on! Would you find these useful? I'm also considering including the constant 'e' and maybe Factorials in the current RBASIC.

See you next time.

see you next time.  
**MICRONICS**  
RESEARCH CORP.

Microcomputers - Hardware and Software  
GIMIX® Sales, Service and Support

33383 LYNN AVENUE, BRITISH COLUMBIA,  
ABbotsFORD, CANADA. V2S 1E2

PS. Special note to Don. Any guidelines you can offer a friend in getting this off the ground? Any pointers would be most appreciated!

*Editor's Note: Bob, I believe that most of the responses you receive will give you enough to stay busy for quite some time. Fact is, I would bet that if you could add all the features asked for, the program would be one big PROGRAM. I mean a big'un!*

However, a BASIC with editor is certainly something users would hanker to. Another is the price; that is important as most users already have a BASIC, so to go to the expense of another one would require that the newer one not only have a lot of neat features, but be within their budget.

So my family here is good. And keep the good thoughts coming.

2200

EOF

Dear Dan,

The "DS-9-6889 Operating System System Programmer's Manual" Section 11.3, 'I/O Service Requests', is the Attach service request description page 11-48, states 'Attach and Detach are like Link and Unlink for devices'. The operating system, however, provides no facility to attach or detach a device directly from the terminal, as the Link and Unlink utilities do for memory modules. Therefore, for completeness two utilities named Attach and Detach, which are analogous to Link and Unlink respectively, are presented here.

But when would the user feel a need to explicitly attach or detach a device? One occasion might be when a device is dynamically installed or removed. Another is given further on in the same paragraph cited above, which states "system performance can be improved slightly if all devices are attached at startup. This increments each device's use count and prevents the device from being reinitialized every time it is opened. This also has the advantage of allocating the static storage for devices all at once, which prevents fragmentation on Level One systems."

To take advantage of this improvement in performance, the system manager may place in the startup file a line such as this:

attach 2010/01/TERM 01/02/03/04/05/06

This shows that, like Link and Unlink, Attack and Detach may act on several devices in a single invocation.

Note that if a device is non-shareable, attaching it in this way will prevent it from being accessed by any other process until it is detached. This is obviously not what is desired, so attach is clearly not useful for non-shareable devices.

S. D. Peters

S. D. Retey

### Run Attach

```

00001          nm  Attach
00002          lli  Attach Devices
00003
00004
00005
00006          ..  .
00007          ..  .
00008          ..  .
00009          ..  Attach
00010          ..  .
00011          ..  .
00012          ..  Created 04/02/09 by G. O. Peters
00013          ..  .
00014          ..  .
00015
00016
00017
00018
00019          *  Standard Label Definitions
00020
00021
00022          *  vco  /DB/DEFS/os9defts
00023          *  b30  /DB/DEFS/os9deflets
00024          *  ripl
00025          *  endc
00026
00027
00028
00029
00030          *
00031          *  Module Definitions
00032          *
00033
00034
00035          0010  MType  equ  Progra  module type:  program
00036          0001  MLang  equ  Objct  module language:  4009 object
00037          0000  MATtr  equ  ReEnt  module attributes:  re-entrant
00038          0001  MRev  equ  1  module revision:  1
00039          0011  MTLn  equ  MLang,MLang module type/language code
00040          0001  MATry  equ  MAtt,MRev module attr/revision code
00041          0200  MStackSz  equ  $0200  module stack size:  2 pages
00042
00043          0000  B7C00020  Module  mod  MSize,MMem,MTLn,MAtr,MRev,MExc,MData
00044

```



```

00095      ****
00096      #      Terminate
00097      ****
00098
00099 0031      Terminal
00100
00101  ****
00102  #      Module Exit
00103  #      ****
00104  #      ****
00105  ****
00106
00107 0031 102686  MExit 007 F0Exit  exit
00108
00109 0034 30A685      mod
00110
00111 0037      MSize  equ  *      module size
00112
00113      end

01010 error(s)
01011 warning(s)
01012 00055 program bytes generated
01216 00312 data bytes allocated
01E43 17747 bytes used for symbols

```

1641 Routt St.  
Lakewood, CO 80215  
September 12, 1986

Don Williams  
'68 Micro Journal  
5900 Cassandra Smith Road  
Hixson, TN 37343

Dear Don,  
I've got some comments about the August and September issues of '68 Micro.

First, regarding Ron Anderson's problem with linefeeds having carriage returns added automatically in OSK (September, p. 10):

This is definitely hard-coded in the SCF file manager (see enclosed disassembly of SCF, page 8, near the bottom). This code is used only by the I\$Writeln system call (writeln() in C), not by the I\$Write system call (write()). What function is Ron calling to send output to the terminal? If he is calling a function which ultimately calls putc() (printf(), puts(), etc.) then he can set a flag to tell putc() whether to use write() or writeln(). (See the C compiler manual - "The Standard Library" chapter - putc()).

Regarding the problem with the OSK C compiler (v. 2.0) stopping with a bus error when processing an initialized float array (August, p. 18):

The C compiler uses a structured variable to represent a constant (it may use it to represent other things [such as variable names] but I don't really want to decipher 300K of assembly language source code to find out). The structured variable contains (among other things) a word which indicates the type of constant, an int which indicates the size of the constant (i.e., 8 for a double, 4 for an int or float) and a location which can hold either a value or a pointer to a value.

When the compiler encounters a floating-point constant (such as "3.14") it converts it from ASCII to a double. It then allocates space for that double, makes the pointer point to the double and sets the size at 8. Or (for those who might follow code better than this explanation):

```

struct structvar *var;
var->dblptr=malloc(sizeof(double));
var->dblptr=&temp_double;
var->size(sizeof(double));

```

If the source code has actually requested a float, then the compiler replaces the pointer to the double with the float equivalent, deallocates the space used by the double, and sets the size to 4:

```

double *tempptr;
tempptr=var->dblptr;
/* tempptr points to double */
*(float *)(&var->dblptr)=tempptr;
/* var->dblptr is now float
   equivalent of double */
var->size(sizeof(float));
free(tempptr);

```

The bus error problem comes a little later when the compiler outputs the assembly source code. It prints the value of the constant (as in "dc.l 0x40000000" for a float 2.) and then deallocates the space used by the structured variable. The deallocating function checks to see if the structured variable refers to a floating-point constant. If it does then the function deallocates the space pointed to by that pointer:

```

if(var->type == FLOATING)
    free(var->dblptr);
    free(var);

```

OOPS! What happens when that "pointer" is actually a float rather than a pointer to a double? Well, free looks near the address of that value to see if it is a valid address. Since there are very few 1 gigabyte OSK systems around (for example - float 2. is 0x40000000 [over 1 billion]) the addressing circuitry will go TILT!

There is a fix for this bug. From the shell type:

```

load debug c68      (if they are not already)
                     (in memory)
debug
l c68
di .r7+df4a a

```

The computer should print (extraneous data omitted here - comments added by me):

```

move.l (e7),a0
cmp.w #75,32(a0) if type is floating-point
bne.s DF66+r7
move.l (a7),a0
tst.l 34(a0)      and pointer is not NULL
beq.s DF66+r7
move.l (a7),a0
move.l 34(a0),d0
bsr E234+r7      then deallocate
move.l (a7),d0      this is DF66+r7

```

If you don't see this code then you have a different version of the compiler and will have to search a disassembly for its equivalent.

If you do see this code type:

```
cu .r7+df54
7000
b0a8
18
660a
2028
22
6704
.
di .r7+df4a a
```

The computer should print:

```
move.l (a7),a0
cmpl.w #75,32(a0) if type is floating-point
bne.s DF66+r7
moveq.l #0,d0
cmp.l 24(a0),d0 and size equals 8 (double)
bne.s DF66+r7
move.l 34(a0),d0
beq.s DF66+r7 and pointer is not NULL
bsr E234+r7 then deallocate
move.l (a7),d0 this is DF66+r7
```

If the computer does not print this code then double-check that you typed the numbers correctly.

If the computer does print this code then type:

```
q
save -rxf=c68 c68 (save c68 to the)
(execution directory)
fixmod c68 -ux (update module CRC)
```

If you haven't received any error messages you should now have a fixed copy of c68.

There are a couple of other bugs in the compiler that you should be aware of.

First, if you have the following code:

```
whatever(argc,arg2)
double arg1;
register int arg2;
```

the compiler thinks arg2 is in register d1. Actually d0-d1 contains arg1 (arg2 is on the stack). This problem does not occur if you don't declare arg2 to be a register. Microware is aware of this bug and responded to my letter to them about it.

The second bug was a little harder to track down for reasons which should become obvious (at least to any machine language programmers).

If your code has complex equations then the compiler usually has to generate code to save any intermediate values - for example:

```
double a,b,c,d,e;
efabs(a-b)-fabs(c-d);
```

In this sample code the program will calculate fabs(a-b), save it to a temporary location, calculate fabs(c-d), subtract it from the previously saved value (i.e. the result of fabs (a-b)) and place the result in e.

The compiler will typically put the temporary value in a pair of registers (such as d4-d5), calculate fabs(c-d) and put it into another pair of registers (d2-d3), move d4-d5 to d0-d1 and do a double subtract call.

However, if the spare data registers (d4-d7) are being used -- for example:

```
double a,b,c,d,e;
register int f,g,h,i;
efabs(a-b)-fabs(c-d);
```

then the compiler will put the temporary value on the stack. Unfortunately, the compiler can become confused about what temporaries are on the stack and what to do with them. The compiler may manipulate the stack in strange ways (particularly after an if statement). If the program takes a certain set of branches then the stack will be at the same level on leaving the function as it was on entering the function. But if the program takes a different branch or set of branches the stack pointer may end up with a different value. This means the computer won't have the correct address to return to when the function is finished.

The program I was running would execute one function hundreds of times with no problem and then crash. Even DEBUG was not much help at first since it displayed only the current value of the program counter rather than keeping track of the last few pc values. After hours of head-scratching, inserting printf statements and working with DEBUG I discovered the stack mismatch and thought "compiler error!". I notified Microware of the problem but never received a reply about it.

This bug can be worked around by not declaring char,short or long variables as registers in a function where the code has to save values temporarily. Pointer variables can be safely declared register since they use the address registers (which are not used for temporary storage).

I want to apologize to those people who wrote to me after reading my letter in the March '86 issue and who are wondering why I haven't responded.

I'm a natural procrastinator (someday I intend to join "Procrastinators Anonymous"). I put off answering those letters for just a little too long so they ended up in a box somewhere in storage when we moved to our new house (actually built in 1914, but new to us). Please write again - your new letters will reach me before I find your old letters!

Sincerely,

*Calvin Dodge*  
Calvin Dodge



**MOTOROLA INC.**

Microprocessor Products Group  
P.O. Box 3800  
Austin, Texas 78764

**EDITORIAL CONTACT:**  
Mark Vercriusse  
912/920-6804

**READER CONTACT:**  
Dean Mosley  
512/440-2839

**INQUIRY RESPONSE:**  
R.Q. Green  
P.O. Box 52073  
Phoenix, AZ 85072

## The MC68030

SECOND GENERATION HIGH PERFORMANCE  
32-BIT MPU

Austin, Texas, September 18, 1986... Motorola's Microprocessor Products Group announces the MC68030, the industry's second generation 32-bit MPU. This enhanced MPU sets the highest performance standards ever for 32-bit MPUs. The MC68030 offers TWICE the performance of the current 32-bit performance leader, the MC68020, while at the same time maintaining 100% upward software code compatibility with the entire MC68000 Family MPUs. The MC68030 started with a high performance MC68020 core and added these performance improvement features: increased internal parallelism, dual on-chip caches with a burst fillable mode, dual internal data and address buses, improved bus interface, and on-chip Paged Memory Management Unit, MMU. The MC68030 MPU is in a unique position to provide the performance required in next generation engineering workstations (2 to 3 times the VAX 8600) as well as the price and functionality necessary to build high volume \$2,000 to \$3,000 office automation systems exceeding the performance of the VAX 8600. The target applications for the MC68030 include:

- \* Office Automation (highest volume)
- \* Engineering Workstations (highest performance)
- \* Fault tolerant computers (transaction processing)
- \* Parallel Processors (supermicrocomputers)
- \* Telephone switching systems (PBX, PABX)
- \* Intelligent Controllers (Communications, Factory Automation, Graphics)

### The MC68030 Introduces "Industry Firsts"

This new high performance second generation 32-bit MPU offers a number of unique features:

- \* FIRST ON-CHIP INSTRUCTION AND DATA CACHES
- \* FIRST ON-CHIP HARVARD STYLE ARCHITECTURE
- \* FIRST DYNAMICALLY CONFIGURABLE BUS INTERFACE
- \* FIRST TRANSPARENT MEMORY WINDOWS

The MC68030 supports the flexible coprocessor interface introduced on the MC68020, and additional pins have been added to the chip to support burst mode, synchronous mode, dual cache support, and emulation providing deterministic tracking of instructions through the pipe and the ability to freeze the contents of the on-chip caches. These features along with the other enhancements added to the performance setting MC68020 core, creates the next 32-bit performance standard -- the MC68030.

### \* FIRST ON-CHIP INSTRUCTION AND DATA CACHES

Today's 32-bit systems implement caches (high speed temporary storage) to increase the performance by supporting the CPU execution unit with immediate access to instructions and data. The MC68020 was the first 32-bit MPU to offer on-chip instruction cache. Now the MC68030 is the first 32-bit MPU to offer both on-chip instruction and data caches. The unique on-chip independent caches provide increased support of the CPU execution unit for higher performance. The dual 256 byte on-chip instruction and data caches boost the data flow to the CPU, typically a major performance bottleneck, to enhance overall throughput. Operating from the caches reduces access times but also reduces the overall bus requirements since the CPU spends less time on the bus accessing data. Bus access time is reduced further by the burst fillable cache mode allowing high speed data fills of both the data and instruction caches. These features increase bus bandwidth for other bus masters including multi-processor systems, LANs, disks, etc. Simulation studies on 17 million bus cycles captured from Unix workstations indicates that the combination of a 256 byte instruction cache and 256 byte data cache is the optimum size when evaluating silicon die size, performance increase, and cache size.

### \* FIRST ON-CHIP HARVARD STYLE ARCHITECTURE

The Harvard style architecture has been around many years in several super and mainframe computers allowing parallel access of data and instructions. Two independent 32-bit address buses and two 32-bit data buses allow the CPU, caches, ROM, and the bus controller to operate in parallel. These parallel instruction and data paths provide the processor an internal bus bandwidth of greater than 80 Mbytes/second. The MC68030 can simultaneously access an instruction from the instruction cache, data from the data cache, and instruction/data from external memory. Optimizing internal parallelism helps the processor achieve the highest performance ever for 32-bit MPUs.

### \* FIRST DYNAMICALLY CONFIGURABLE BUS INTERFACE

The MC68030 will find its way into many types of systems from low-end (\$2,000-\$3,000) to high performance (>\$30,000) with much of the system tradeoffs taking place in the memory subsystem. Low end systems will typically interface directly to low cost DRAM using only the internal caches for performance increases while high performance systems will use a multi-level hierarchy of memory with high speed cache coupled to the MC68030 which then may access a range of memories from high speed static to slow dynamic RAMs.

To support these features the MC68030 supports both a synchronous bus interface (with a minimum 2 clock access) which allows maximum access time to a cache subsystem as well as an asynchronous interface (as exists on the MC68020) for slower memories, peripherals, and other MC68020 compatible subsystems. This interface supports synchronous or asynchronous accesses on a cycle by cycle basis as determined by the memory subsystem requested. Like the MC68020, compatibility is maintained by supporting the existing asynchronous dynamic bus timing feature which allows interfaces to 8-, 16-, or 32-bit devices.

Additionally, this bus interface provides the low end systems designer increased performance by taking advantage of the shorter access times offered by the paged mode, nibble mode, static column DRAM technology. Here, the MC68030 can request a burst fill to the internal caches. Since datum in the cache is organized in rows of four longwords, up to three additional longwords may be loaded during the access. The MC68030 will request a burst fill and the system can then, in as little as one clock per subsequent access, supply the MC68030 with the successive data obtained from the memory.

#### \* ADDITIONAL PERFORMANCE ENHANCEMENTS

Additional gains in performance are made by bringing the function of memory management on-chip with a Harvard style architecture. The dual bus structure offers 80 Mbytes of bandwidth and allows the MMU to be efficiently integrated in-chip. In doing this, the time to translate logical addresses to physical addresses can be hidden during cache access, so that the system will see no performance degradation due to MMU translations. Additionally, the on-chip MMU is coupled to the instruction and data caches so that accesses to the on-chip caches are performed in parallel and the MMU is not utilized unless required.

The MMU portion of the MC68030 provides a high-powered set of functions available on the MC68851 Paged Memory Management Unit to include multiple page sizes, multi-level translation trees, on-chip Address Translation Cache (ATC), and automatic access history maintenance. The MMU on the MC68030, like the MC68851, will automatically search the main memory for address translations when they are not found in the ATC.

The on-chip MMU reduces the minimum physical bus cycle time to two clocks, 1/2 the time required by the MC68020 and MC68851. Internal pipelining permits the MMU to add NO translation time to any bus cycle; physical accesses are just as fast as logical accesses. The 22-entry fully associative on-chip Address Translation Cache (ATC), helps maximize performance. ATC hit rates will be greater than 99% for 4K pages and about 98% for 1K pages.

The MC68030 supports the same powerful coprocessor interface inherent to the MC68020 MPU, MC68881 Floating Point Coprocessor (FPCP), MC68882, the new Enhanced FPCP, and the MC68851 PMMU. Unique dynamic disabling of the address translation supports off-chip MMUs as well as emulation support when translations must be disabled for the emulation software to execute.

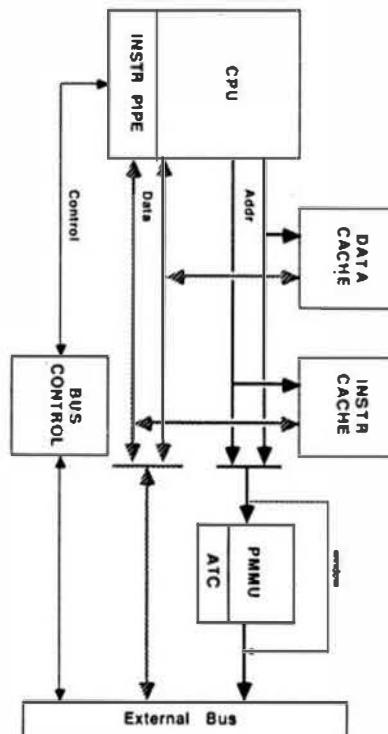
Occasionally, a system may require references to memory locations that cannot afford the time required to search tables for a correct translation. For example, if the MC68030 is sharing the graphics function in a low cost system, drawing a line across the screen should appear continuous. Should a translation in the ATC miss, this line drawing may appear erratic as the tables are searched.

To eliminate this problem, the MC68030 provides the feature of "transparent memory windows" which the application can define to map directly through the logical to physical address space bypassing the MMU. In this way, there is no overhead incurred for time critical portions of the application. These windows can be as small as 16 Mbytes or as large as the entire 4 Gbyte address space supported by the MC68030.

#### MC68030 Availability

The MC68030, designed in Motorola's 1.2 micron HCMOS single-layer metal with silicide process at 10 MHz operation will see first silicon in March, 1987. This high density MPU contains an effective transistor count of approximately 300,000 on a chip of approximately 378 mils on a side. The MC68030 32-bit MPU will be general sampled in July, 1987, in a 128 lead PCA packages at 16.67 MHz. Production will begin October, 1987. The MC68030 MPU will also be incorporated into a high-performance VMEbus microcomputer board to be available in the fourth quarter 1987. Development system support will also be provided in 1987.

#### MC68030 BLOCK DIAGRAM





## MOTOROLA INC.

Microprocessor Products Group  
P.O. Box 3600  
Austin, Texas 78764

EDITORIAL CONTACT:  
Mark Verbrugge  
512/928-6804

READER CONTACT:  
Dean Mosley  
512/440-2839

INQUIRY RESPONSE:  
R.Q. Green  
P.O. Box 52073  
Phoenix, AZ 85072

### MOTOROLA ANNOUNCES THE MC68882 ADVANCED FLOATING POINT COPROCESSOR

Austin, Texas, September 18, 1986... Motorola's Microprocessor Products Group announces the MC68882, the Industry's second generation 32-bit Floating Point Coprocessor. The Enhanced Floating Point Coprocessor (EFPCP) offers TWO to FOUR times the performance of the MC68881, the first Floating Point Coprocessor to strictly conform to 754, IEEE Standard for Binary Floating Point Arithmetic. The new MC68882 EFPCP offers the same strict conformance to the IEEE Standard, plus software compatibility, pin compatibility, and broad based functionality.

The EFPCP offers the basic math functions: add, subtract, multiply, and divide. Unlike any competitive unit, the MC68882 goes much further and provides a full selection of transcendental and non-transcendental functions. These operations include root values, trigonometric functions, exponentials, hyperbolics, and logarithms.

#### FEATURES

##### The MC68882 features:

- Eight general purpose floating-point data registers, each supporting a full 80-bit precision real data format (64-bit mantissa plus a sign bit, and a 15-bit signed exponent).
- A 67-bit arithmetic unit to allow very fast calculations, with intermediates precision greater than the extended precision format.
- A 67-bit barrel shifter for high-speed shifting operations (for normalizing, etc.).
- Forty-six instructions, including 35 arithmetic operations.
- Full conformance to the IEEE 754 standard, including all requirements and suggestions.
- Twenty-two constants available in the on-chip ROM, including Pi, e, and powers of 10.
- Operation with any host processor, on an 8, 16, 32-bit data bus.

In order to increase performance over the MC68881, a Conversion Control Unit (CCU) was added to the MC68882. The CCU improves the performance of the FMOVE instruction and most of the arithmetic operations by speeding up the most common binary data format conversions (conversions between the internal 80-bit data format and single, double, and extended precision formats used externally). The CCU also directs the

communication dialog concurrently with the activity of the Execution Control Unit (ECU). This allows the MC68882 to pipeline the execution of multiple floating-point instructions with the MC68020 or MC68030 increasing internal parallelism and boosting performance. Also, the floating-point data register array is dual ported so that it can be accessed simultaneously by the CCU and ECU to allow fully concurrent execution of the ECU and the CCU supporting concurrent loading, storing, and computation. This increased parallelism further boosts the performance.

#### PERFORMANCE

From the standpoint of applications software, the MC68882 and the MC68881 are identical, right down to the results that are generated by calculations. The instruction set supported by both devices is also the same, so that systems that utilize the MC68881 can get about a 50% performance increase by simply unplugging the MC68881 and replacing it with an MC68882. For those designs that optimize the software, a 2 X to 4 X improvement in performance can be expected.

current 16 MHz MC68881 customers have measured over 1 Million Whetstones performance, and at 20 MHz 1.7 Million Whetstones have been reported. Simply replacing a MC68881 with a MC68882 will provide up to 1.7 Million Whetstones. For those designs that optimize the software, the MC68882 will perform over 3.5 Million Whetstones.

The MC68882 EFPCP is a high performance, single-chip, NMOS VLSI device designed to operate primarily as a coprocessor with the high performance MC68020 32-bit MPU and the new 32-bit performance standard, the MC68030 MPU. The EFPCP is closely coupled to the MC68020 MPU or the MC68030 MPU through the MC68000 Coprocessor Interface, a standard feature of these high performance devices. The MPU and the EFPCP share the tasks of interconnect. The Coprocessor Interface is transparent to the systems programmer, as coprocessor instructions are written as part of the main program instruction stream. The MPU passes coprocessor instructions to the MC68882 which operates concurrently with the main processor, thereby freeing the CPU for other tasks.

In addition to the closely coupled coprocessor architecture with the MC68020 and MC68030 MPUs, the MC68882 can be used with any of the MPU devices of the M68000 Family, and it may also be used as a peripheral to non-M68000 processors.

#### AVAILABILITY

The MC68882 designed in Motorola's 1.5 micron NMOS process for 20 MHz operation will see first silicon December 1986. This high density device contains an effective transistor count of approximately 168,000 on a chip of approximately 287 mils on a side. This NMOS device will be general sampled in April 1987 in a 68 lead pin grid array package at 16.67 MHz. The MC68882 is pin-to-pin compatible and timing is identical to the MC68881 FPCP. Production will begin in August 1987.



MICROWARE SYSTEMS CORPORATION

## NEWS RELEASE

Microware Systems Corporation  
Andrew Crane  
515-224-1929

DATE: September 8, 1986

SUBJECT: OS-9 AVAILABLE ON FORCE VME BUS PROCESSOR BOARDS

## FOR IMMEDIATE RELEASE

Des Moines, IA -- Force Computers GmbH has signed a license with Microware Systems Corporation for the distribution of Microware's OS-9/68000 Operating System. Under the agreement, Force will be able to offer new and existing users of its family of VME-based 68000 processor boards and peripherals with OS-9.

Initial versions of OS-9/68000 will be available for Force's SYS68K/CPU-20/21 and SYS68K/CPU-4V processors plus drivers for the SYS68K/NPC-1 floppy/hard disk controller. OS-9 support for other Force peripheral boards can be developed by Force Systems users utilizing Microware's PortPak Development package.

Force VME users will now be able to capitalize on Microware's modular operating system, and powerful development languages and tools; including Microware Basic, Pascal, C and Fortran, plus the growing base of OS-9 application programs available from third party vendors.

## OS-9 AVAILABLE ON FORCE VME BUS PROCESSOR BOARDS

This is the latest in a series of announcements reflecting OS-9's rapidly increasing acceptance as the standard operating system for 68000 based computers. Earlier this year Philips and Sony announced OS-9/68000 will be the basis for the Compact Disc - Interactive (CD-I) standard. In July, Thomson, Olivetti and Acorn signed an agreement to cooperate in the development of a European Standard for 16 bit microcomputers incorporating OS-9/68000. In late August, Tandy released the Color Computer 311 based on Microware's operating system and graphics user interface.

OS-9/68000 is a real-time, multi-user, multi-tasking operating system for computers based on the 68000 family of microprocessors. It is compact, ROMable and provides a UNIX-style environment for application software. Since its introduction in 1983, OS-9/68000 has been licensed to over 250 manufacturers for use in a wide variety of industrial, scientific and consumer products.

Founded in 1977, Microware specializes in the development of advanced 68000 family operating systems and programming languages. Microware offices are located in Des Moines, Iowa and Tokyo, Japan with field representatives worldwide.

North American Amiga Users Group  
Richard Shoemaker  
Box 376  
Lemont, PA 16851

To: Computer related publications

MEMO: FOR IMMEDIATE RELEASE

re: New Telecommunications Resource

The North American Amiga Users Group (NAAUG) is proud to announce it has established a 24-hour BBS to support NAAUG members. The board will also grant limited access to non-member Amiga users. The number is (814) 339-6042.

One of the primary purposes of the BBS (named the 'LifeSaver') is to provide a central point for the collection and distribution of Amiga Public Domain and User Supported software. The BBS is also designed to provide a place for users to share problems, ideas, tips and to buy or sell personal computer equipment. NAAUG's newsletter, "AmigaHelp", will be published electronically, concurrently with the printed version.

NAAUG will send a sample copy of "AmigaHelp" to any one who requests one, by mail or by leaving a message on the BBS.

NAAUG is the one of the world's largest Amiga User Groups. It is organized to support users of Amiga computers, Epson printers, and all PC compatible computers. Annual membership is only \$25; and includes a subscription to "AmigaHelp" a free disk of public domain software, participation in a money saving Co-Op, free classified ads, and access to over 50 disks of public domain software. NAAUG can be contacted at: Box 376, Lemont, PA 16851 (814) 237-5511 after 4 PM until 9 PM and weekends (voice only).

\*\*\*

From the date of:  
Barry Balitski  
151 Midglen Place  
Calgary, Alta.  
Canada T2X 1H6

Dear Mr. Williams

I have sent in two articles previously which I was delighted to see were published. The articles were **FLEX PRINTER SPOOLER CONTROL** published Feb. 1986 and **BAD MEMORIES** May 1986. It was delightful to see my name in print and thank you for the subscription extension. I really like the new motto I have seen in my latest issue 'contribute nothing - expect nothing' as we are the people who must all work to keep the magazine alive. Therefore I enclose my latest writing effort which I hope you will give your consideration for publishing. I have enclosed the article on hardcopy conforming to the 4.5 inch article width. I also enclose a diskette formatted 88 SD 35 track FLEX. Please use whichever you find most convenient. Don't worry about returning the diskette as it's probably not worth the return postage to Canada.

I would also like to express my appreciation to all the staff at '68 MJ' for the fine job they do in producing the only computer magazine I read several times cover to cover.

I thank you for your consideration in putting my name in print.

*BBalitski*  
Barry Balitski

MAGNON ENGINEERING, INC.  
The leader in E.M.F. commutated (closed loop) stepper motor drives.

Magnon Engineering announces the release of their E.M.F. commutated stepper motor drives and controllers. The heart of these units is the 6809 microprocessor. These cards presently come in the following buses - Std, Multi, Eurocard G64 & 96 and coming soon in an IBM bus.

The advantages of using a commutated stepper motor drive are across the board.

- \* Higher speeds.
- \* Less power to accomplish the same task.
- \* Smaller and less expensive motors to perform same task.
- \* No external feedback is required to keep position.
- \* Board contains both driver and controller.
- \* The base cost of Magnon card is considerably less than open loop systems.
- \* Bus level Product
- \* Saving of dollars on power supply, motors and feedback system.

## Disadvantages

- \* An initial set up will be required for those who do not purchase a motor from Magnon.
- \* In order for E.M.F. commutation to occur, the first initial starting steps will have to be taken open loop. This is programmable so the user knows just how many steps have been taken.

These disadvantages have been eliminated by using the Magnon 12000 Intelligent Programmable Motion Controller and by giving Magnon Engineering your load requirements and purchasing your complete controllers with motors already installed for your system.

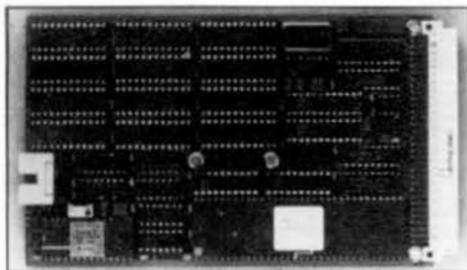


#### LOW COST, HIGH RESOLUTION GRAPHICS CONTROLLER RIDES THE G-64 BUS

AKAHIM, CA, WESCON '86, BOOTH 3167, NOVEMBER 18, 1986—LESPAC introduces a new high performance graphics controller built on a single height Eurocard, compatible with the G-64 bus. The GESVIG-4 is fully programmable and can display up to 640 by 480 pixels on a non-interlaced screen. The GESVIG-4 allows up to 256 different colors out of a choice of 262,144 to be simultaneously displayed on the screen.

The GESVIG-4 uses an advanced CRT controller capable of drawing speeds of up to 2.5 million pixels per second. The board can also handle advanced functions such as vector and circle drawing, pattern fill and scaling. The GESVIG-4 also supports windows to be created and moved within the display area.

The GESVIG-4 is built on a Eurocard of 100 by 160 millimeters and is compatible with the standard G-64 bus. The G-64 bus is an easy-to-interface, non-multiplexed, 16-bit bus. The resulting stiffness of the small board format, and the high reliability of the DIN connector, makes the G-64 bus ideal for a variety of low to mid performance industrial applications.



Because of its very compact form factor of 25 square inches, its low cost and high performance, the GESVIG-4 is ideally suited for such applications as industrial data terminals, navigation computers, and graphics workstations.

The GESVIG-4 is supported with a VDI compatible software driver interface. The driver, referred as GESGPS-3, is designed to operate in Microware's OS-9, 68000 operating system environment.

The GESVIG-4 is available today for the low unit price of \$1250. Deliveries for low quantities are from stock. Production order deliveries are 4 to 8 weeks. GESGPS-3 is available today for \$250 per copy.

For more information contact: Joe Murphy  
LESPAC, Inc.  
100 W. Hoover Ave.  
Mesa, AZ 85202  
(602) 962-5559

# SK•DOS™

The Generic DOS™ for 68000 applications in

- ★ Industrial Control
- ★ Business Use
- ★ Educational Computing
- ★ Scientific Computing
- ★ Number Crunching
- ★ Dedicated Systems
- ★ Turnkey Systems
- ★ Data Collection
- ★ Single-board Computers
- ★ Bus oriented Computers
- ★ Graphic Workstations
- ★ One-of-a-kind Systems
- ★ Advanced Hobbyist Use

SK•DOS is a single user disk operating system for computers using Motorola 32 bit CPUs such as the 68008, 68000, 68010, and 68020. It provides the power of a full DOS, yet is simple and easy to use, and will run on systems from 32K to 16 megabytes. Because SK•DOS is easily implemented on a new system, we call it "The Generic DOS" which allows programs written for one system to be run on many others.

SK•DOS comes with over 40 commands and system programs, including a 6809 emulator which allows 68K SK•DOS to run application programs and languages developed for 6809 SK•DOS and other systems. Assemblers, editors, and higher level language support are available from third party software vendors and through public domain software.

SK•DOS is available for single-copy or dealer sales, as well as OEM licensing. Single copies cost \$125 (inquire as to available systems). Extremely attractive OEM licensing terms are also available. An optional Configuration Kit contains a detailed Configuration Manual and two disks of source code for system adaptation, including source code for a system monitor/debug ROM and other programs useful for adapting SK•DOS to new systems.

## SK•DOS™

(as well as other fine 68000 and 6809 hardware and software products)

is available from



## PLEASE NOTE...

Effective immediately,  
we have officially changed  
our name from



to





**Now!** "TOPS  
The OFFICE PRINT Shop™" Makes professionals of us all.  
For less than just affordable!

# DeskTop Publishing

**NOTE:** The following includes a 3 day, *hands-on*, instructional session for the entire "The OFFICE PRINT Shop™" system. A full 6 months, *no extra charge*, telephone or in-house (our offices, normal business hours) follow-up advisory service. We feature full Apple™ service and also national service by Honeywell. This includes *next day*, on site service. Over 95% of all service requests are completed on the initial call. *We understand the importance of fast service!*

100

**System 100** *Very Affordable Save up to 90% on your printing*

The TOPS System 100

consists of the following items:

- A special Apple Macintosh Plus™ 1 Megabyte computer, including a double sided, double density 800,000+ character disk drive.
- An Apple LaserWriter™ typeset quality printer.

*Page make-up software  
galley typesetting software  
3 day - on site - instructions  
6 months instructional support &  
much more!*



**Option:**

We also offer software to drive most all the large commercial typesetters. You can save a bundle by doing your own typesetting and proofing, and then downloading to the commercial system.

Leasing with payout available for all systems.



Also available with 20 million character storage 'Hard Disk'

## Data-Comp Division

CP



A Decade of Quality Service'

Systems World-Wide

Computer Publishing, Inc. 5900 Cassandra Smith Road  
Telephone 615 842-4601 - Telex 510 600-6630 Hixson, Tn 37343

This document composed, typeset and printed with a TOPS System 100

TOPS The OFFICE PRINT Shop is a trademark of Computer Publishing, Inc.

Apple Macintosh Plus is a trademark of Apple Computer Company, Inc.

LaserWriter is a trademark of Apple Computer Company, Inc.



## OS-9 UniFLEX MUSTANG-020, 68020, 68881 AND MORE HANDS-ON EXPERIENCE

The DATA-Comp Division of Computer Publishing Corporation announces their new and innovative HANDS-ON 68020 computer familiarization two day event. A chance to TRY BEFORE YOU BUY!

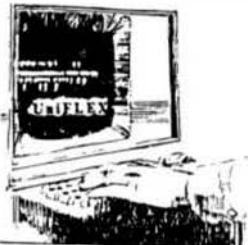
For two full days (Monday through Friday - excluding legal holidays) each participant will be furnished the exclusive use of a 68020 computer (MUSTANG-020). Each system will have available native C compilers, BASIC, assembler and other high level languages. Each system will be equipped with the Motorola MC 68881 math co-processor, where applicable.

Each demonstration room will contain not more than two work stations. Each system will be equipped with floppy disk, 20 megabyte Winchester technology hard disk, and 2 megabyte of RAM. RAM is partitioned as 690K bytes of RAM disk and 1.2 megabyte of user RAM space.

Participants are encouraged to bring along any source level projects, for evaluation, in C, BASIC or assembler. Call for availability of other HHLs.

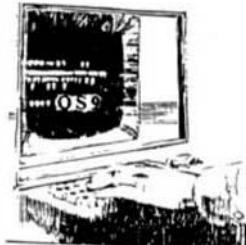
Although this is not a training seminar, Data-Comp personnel are available for assistance and consultation. This event is scheduled for hands-on evaluations of the 68020 CPU, 68881 math co-processor and MUSTANG-020 system, operating in a functional environment.

Transportation to and from the airport and hotel/motel will be provided. Lunch provided both days. Chattanooga airport is serviced by American, Delta, Republic and other airlines.



**COST**  
One person - \$375.00  
Two persons - \$595.00

\* Motel single \$22.00, double \$26.00  
Includes satellite TV - convenient to food and shopping



### DATA-COMP

*A Division of  
Computer Publishing, Inc.*

5900 Cassandra Smith Road  
Hixson, Tn 37343  
Telephone 615 842-4600  
Telex 510 600-6630

Systems available for both OS-9 and UniFLEX. Reservation should be made 15 days in advance. Attendee should initially indicate OS-9, UniFLEX or both. Special facilities available on request. Please write or call for additional information.

NOTE: Both OS-9 and UniFLEX are Unix type operating systems. Each as been enhanced in some aspect or another. Prospective attendees should have some working knowledge or experience with one of these operating systems, to gain full benefit of the session. However, a newcomer will find that it is a simple matter to be fairly proficient in using these systems in the allocated time. Special system instruction available on request. Call or write.

\* Hotel/Motel cost are separate cost, not included in the basic cost shown.

# THE 6800-6809 BOOKS

..HEAR YE.....HEAR

## OS-9™ User Notes

By: Peter Dibble

The publishers of 68' Micro Journal are proud to make available the publication of Peter Dibble's OS9 USER NOTES

Information for the BEGINNER to the PRO,  
Regular or CoCo OS9

**Using OS9**  
HELP, HINTS, PROBLEMS, REVIEWS, SUGGESTIONS, COMPLAINTS,  
OS9 STANDARDS, Generating a New Bootstrap, Building a  
new System Disk, OS9 Users Group, etc.

**Program interfacing to OS9**  
DEVICE DESCRIPTORS, DIRECTORIES, "FORKS", PROTECTION,  
"SUSPEND STATE", "PIPES", "INPUT/OUTPUT SYSTEM", etc.

**Programming Languages**  
Assembly Language Programs and Interfacing; Basic09, C,  
Pascal, and Cobol reviews, programs, and uses; etc.

**Disk Include**  
No typing all the Source Listings in. Source Code and, where applicable, assembled or compiled Operating Programs. The Source and the Discussions in the Columns can be used "as is", or as a "Starting Point" for developing your OWN more powerful Programs. Programs sometimes use multiple Languages such as a short Assembly Language Routine for reading a Directory, which is then "piped" to a Basic09 Routine for output formatting, etc.

**BOOK \$9.95**

Typeset -- w/ Source Listings

(3-Hole Punched; 8 x 11)

Deluxe Binder - - - - - \$5.50

### All Source Listings on Disk

1-8" SS, SD Disk - - - \$14.95  
2-5" SS, DD Diska - - - \$24.95

Shipping & Handling \$3.00 per Book, \$2.50 per Disk set

Foreign Orders Add \$4.50 Surface Mail  
or \$7.00 Air Mail

If paying by check - Please allow 4-6 weeks delivery  
\* All Currency in U.S. Dollars

**Continually Updated In 68 Micro Journal Monthly**

Computer Publishing Inc.  
5900 Cassandra Smith Rd.  
Hixson, TN 37343



\*FLEX is a trademark of Technical Systems Consultants  
\*OS9 is a trademark of Microware and Motorola  
\*68' Micro Journal is a trademark of Computer Publishing Inc.



## FLEX™ USER NOTES

By: Ronald Anderson

The publishers of 68 MICRO JOURNAL are proud to make available the publication of Ron Anderson's FLEX USER NOTES, in book form. This popular monthly column has been a regular feature in 68' MICRO JOURNAL SINCE 1979. It has earned the respect of thousands of 68 MICRO JOURNAL readers over the years. In fact, Ron's column has been described as the 'Bible' for 68XX users, by some of the world's leading microprocessor professionals. The most needed and popular 68XX book available. Over the years Ron's column has been one of the most popular in 68 MICRO JOURNAL. And of course 68 MICRO JOURNAL is the most popular 68XX magazine published.

Listed below are a few of the TEXT files included in the book and on diskette.

All TEXT files in the book are on the disks.

LOGOC1	File load program to offset memory — ASM PIC
MEMOVEC1	Memory move program — ASM PIC
DUMP C1	Printer dump program — uses LOGO — ASM PIC
SUBTEST C1	Simulation of 6800 code to 6809, show differences — ASM
TERMEM C2	Modem input to disk (or other port input to disk) — ASM
MC2	Output a file to modem (or another port) — ASM
PRINT C3	Parallel (enhanced) printer driver — ASM
MODEM C2	TTL output to CRT and modem (or other port) — ASM
SCIPKG C1	Scientific math routines — PASCAL
UC4	Mini-monitor, disk resident, many useful functions — ASM
PRINT C4	Parallel printer driver, without PFLAG — ASM
SET C5	Set printer modes — ASM
SETBAS1 C5	Set printer modes — A-BASIC

NOTE: .C1..C2, etc.=Chapter 1, Chapter 2, etc.

\*\*Over 30 TEXT files included in ASM (assembler)-PASCAL-  
PIC (position independent code) TSC BASIC-C, etc.

Book only: \$7.95 + \$2.50 S/H

With disk: 5" \$20.90 + \$2.50 S/H

With disk: 8" \$22.90 + \$2.50 S/H

(615) 842-4601  
Telex 5108006630

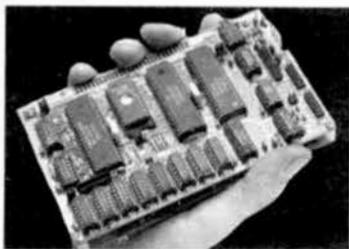
# SK \* DOS

(formerly called STAR-DOS)  
is now available for both  
**68000 and 6809**

computers. The same great DOS, but now better than ever, with enhancements which make it ideal for 6809 users moving to the 68000/68008/68010/68020. Available off-the-shelf now for the Emerald ESB-1 and Peripheral Technology PT-68K, and for licensing to OEMS at attractive terms. Single copies to end users are \$75 (6809 version) and \$125 (68K version). Configuration Manual (optional at \$50) gives full details on adapting to new systems, supplied FREE to SK\*DOS/68K purchasers until Dec. 1. Adapt SK\*DOS to a new system and receive a royalty on your adaption! Call us at 914-241-0287 for more information.



Box 209 Mt. Kisco NY 10549



## 512K RAM Expansion

## Compact Flexible 6809 Computer

The new ST-2900 system — a complete 64K small business or hobbyist computer is only one of its many possible configurations. Among its features are:

- Small enough to hold in your hand! (Eurocard size: 3.9" x 6.3")
- Three board "system" for greater versatility than single board computers.
- CPU Board — powerful 6809E processor, 16K or 64K RAM, 1K-32K EPROM, 2 RS232 serial ports with software programmable baud rates, 16 bit counter/timer. Run the CPU board all by itself, or plug your own custom board or our FDC board and/or RAM-512 board into the expansion connector.
- FDC Board — double-sided/double-density floppy disk controller with adjustment free digital data separator and write precompensation, 2 8-bit parallel ports, 2 16-bit counter/timers, prototyping area.
- RAM-512 Board — 524,288 bytes of RAM on a 4.15" x 6.3" board. Low power. Includes RAM-Disk software for FLEX/STAR-DOS or OS-9.
- FLEX, STAR-DOS, and OS-9 supported — software selectable.
- OS-9 Conversion Package lets you use the low cost Radio Shack CoCo version of OS-9 on our ST-2900 system. Save \$131 off the suggested list price of OS-9! **No programming is involved.** Supports CoCo OS-9, standard OS-9, and MIZAR OS-9/68K disk formats. Compatible with PC-XFER to let you read/write/format MS-DOS disk!
- CPU bare board plus EPROM \$45 OS-9 Conversion Package \$49
- FDC bare board \$38 FLEX Conversion Package \$29
- RAM-512 board A&T (w/o RAM) \$299 CPU + FDC + OS-9 Conversion \$119
- CPU + FDC board set assembled and tested \$329
- Add \$5 shipping/handling (\$10 overseas). These prices are in U.S. funds. Canadian orders: call or write for prices. Terms: check, money order, VISA.

Trademarks FLEX — Technical Systems Consultants, OS-9 — Microware & Motorola, MS-DOS — Microsoft



**SARDIS**  
TECHNOLOGIES

2261 E. 11th Ave. Vancouver, B.C., Canada V5N 1Z7

Call or write for free catalog  
and complete price list  
(604) 255-4465

Source for ST-2900 OS-9 device drivers now available

# SPECIALTY ELECTRONICS

## S-O-F-T-W-A-R-E

for  
**\* OS9/68000**

### ACCOUNTING

Accounts Receivable .....	\$399
Accounts Payable .....	\$399
General Ledger/Cash Journal .....	\$499
Payroll .....	\$499
Inventory Control .....	\$499

### UTILITIES

Disk Repair .....	\$79.95
Modem 68K .....	\$79.95
Chgattr .....	\$19.95
Dirs .....	\$19.95



For more information contact  
**Specialty Electronics**  
909 N. Cleveland / Enid, Oklahoma 73703  
(405) 233-1632

\* trademarks OS9 Microware, Inc.

## Hard Disk Subsystem for SS-50 Computers

THIS PROVEN SUBSYSTEM ADDS HARD DISK SPEED AND STORAGE CAPACITY TO YOUR COMPUTER YET REQUIRES ONLY ONE SS-50 SLOT. SOFTWARE (WITH SOURCE) IS INCLUDED FOR YOUR CHOICE OF FLEXY OR SK\*DOS\*, OS-9 LEVEL I OR LEVEL II, OR OS-9 68K OPERATING SYSTEMS. THE SOFTWARE HONORS ALL OPERATING SYSTEM CONVENTIONS. THE SOFTWARE IS DESIGNED FOR THE XEBEC S1410 CONTROLLER INTERFACING TO ANY HARD DISK DRIVE THAT CONFORMS TO THE ST506 STANDARD. FOUR SUBSYSTEMS ARE AVAILABLE:

- 1) 27 MB (FORMATTED) CONTROL DATA CORPORATION WREN HARD DISK, XEBEC S1410 CONTROLLER, SS-50 INTERFACE CARD, ALL CABLES, AND SOFTWARE FOR \$2850;
- 2) 7.3 MB (FORMATTED) TANDON TM-603 HARD DISK, REST SAME AS ABOVE FOR \$895;
- 3) NO HARD DISK, REST SAME AS ABOVE FOR \$600; AND
- 4) SS-50 INTERFACE CARD AND SOFTWARE FOR \$200.

ALL PRICES INCLUDE SHIPPING. WE ACCEPT VISA AND MASTERCARD WITHOUT ADDING A SURCHARGE. TEXAS RESIDENTS MUST ADD SALES TAX. THE SUBSYSTEM MAY BE MOUNTED WITHIN YOUR COMPUTER CHASSIS OR IN A SEPARATE ENCLOSURE WITH POWER SUPPLY. PLEASE WRITE OR PHONE (INCLUDE YOUR DAY AND EVENING PHONE NUMBERS) FOR MORE INFORMATION. WE WILL RETURN NORTH AMERICA CALLS SO THAT ANY DETAILED ANSWERS WILL BE AT OUR EXPENSE.

**HELLWRITTEN**  
ENTERPRISES



P.O. Box 9802 - 845  
AUSTIN, TEXAS 78766

\*\*\* (512) 244-6530 \*\*\*



FLEX is a trademark of Technical Systems Consultants, Inc.  
SK\*DOS is a trademark of Star-Kits  
OS-9 is a trademark of Microware and Motorola

## SOFTWARE FOR 680x AND MSDOS

### SUPER SLEUTH DISASSEMBLERS

EACH \$99-FLEX \$101-OS/9 \$100-UNIFLEX

OBJECT-ONLY versions: EACH \$50-FLEX, OS/9, COCO

interactively generate source on disk with labels, include xref, binary editing  
specify 6800, 1, 2, 3, 5, 8, 9/6502 version or Z80/8080, 5 version  
OS/9 version also processes FLEX format object file under OS/9  
COCO DOS available in 6800, 1, 2, 3, 5, 8, 9/6502 version (not Z80/8080, 5 only)  
NEW: 68010 disassembler, \$1 .FLEX, OS/9, UNIFLEX, OS/9-68K, MSDOS

### CROSS-ASSEMBLERS WITH MACRO CAPABILITIES

EACH \$50-FLEX \$100-OS/9 \$100-UNIFLEX

specify for 180x, 6502, 6801, 6804, 6805, 6809, Z80, 8048, 8051, 8085, 6800  
modular cross-assembler in C, with load/unload utilities NOW: OS/9-68K  
8-bit (not 68000) sources for additional \$50 each, \$100 for 3, \$300 for all

### DEBUGGING SIMULATORS FOR POPULAR 8-BIT MICROPROCESSORS

EACH \$75-FLEX \$100-OS/9 \$80-UNIFLEX

OBJECT-ONLY versions: EACH \$50-COCO FLEX, COCO OS/9  
interactively simulate processor, include debugger, memory, binary editing  
specify for 6800, 1, 2, 3, 5, 8, 9/6502, 6809 OS/9, Z80 FLEX

### ASSEMBLER CODE TRANSLATORS FOR 6502, 6800/1, 6809

6502 to 6809 \$75-FLEX \$85-OS/9 \$40-UNIFLEX

6800/1 to 6809 & 6809 to position-ind. \$50-FLEX \$75-OS/9 \$60-UNIFLEX

### FULL-SCREEN X BASIC PROGRAMS with cursor control

AVAILABLE FOR FLEX, UNIFLEX, AND MSDOS

DISPLAY GENERATOR/DOCUMENTOR	\$50 w/source, \$25 without
MAILING LIST SYSTEM	\$100 w/source, \$50 without
INVENTORY WITH MRP	\$100 w/source, \$50 without
TABULA RASA SPREADSHEET	\$100 w/source, \$50 without

### DISK AND X BASIC UTILITY PROGRAM LIBRARY

\$50-FLEX \$30-UNIFLEX/MSDOS

edit disk sectors, sort directory, maintain master catalog, do disk sorts,  
resequence some or all of BASIC program, read BASIC program, etc.  
non-FLEX versions include son and resequence only

### CMODEM TELECOMMUNICATIONS PROGRAM

\$100-FLEX, OS/9, UNIFLEX, MS-DOS, OS/9-68K, UNIX

OBJECT-ONLY versions: EACH \$50  
menu-driven with terminal mode, file transfer, MODEM7, XON-XOFF, etc.  
for COCO and non-COCO, drives internal COCO modem port up to 2400 Baud

## DISKETTES & SERVICES

### 5.25" DISKETTES

EACH 10-PACK \$12.50-SSSD:SSDD/DSDD

American-made, guaranteed 100% quality, with Tyvek jackets, hub rings, and labels

### ADDITIONAL SERVICES FOR THE COMPUTING COMMUNITY

#### CUSTOMIZED PROGRAMMING

We will customize any of the programs described in this advertisement or in our  
brochure for specialized customer use or to cover new procedures; the charge  
for such customization depends upon the marketability of the modifications.

#### CONTRACT PROGRAMMING

We will create new programs or modify existing programs on a contract basis.  
a service we have provided for over twenty years. the computers on which we  
have performed contract programming include most popular models of  
mainframes, including IBM, Burroughs, Univac, Honeywell, most popular  
models of microcomputers, including DEC, IBM, DG, MP, AT&T, and most  
popular brands of microcomputers, including 6800/1, 6809, Z80, 6502  
68000, using most popular languages and operating systems on platforms  
ranging in size from large telecommunications to single board controllers;  
the charge for contract programming is usually by the hour or by the task.

#### CONSULTING

We offer a wide range of business and technical consulting services, including  
advice, training, and design, on any topic related to computers;  
the charge for consulting is normally based upon time, travel, and expense.

Computer Systems Consultants, Inc.  
1454 Letta Lane, Conyers, GA 30207  
Telephone 404-483-4570 or 1717

We take orders at any time, but plan  
long discussions after 6, if possible.

Contact us about catalog, dealer, discounts, and services.

Most programs in source; give computer, OS, disk size.  
25% off multiple purchases of same program on one order.  
VISA and MASTER CARD accepted; US funds only, please.  
Add GA sales tax (if in GA) and 5% shipping.

(UN)FLEX fm Technical Systems Consultants, OS/9 Microware, COCO Tandy/MSDOS Microsoft

# SOFTWARE FOR HARDCORE

• FORTH PROGRAMMING TOOLS from the 68XX&X •  
• FORTH specialists — get the best!! •

NOW AVAILABLE — A variety of rom and disk FORTH systems to  
run on and/or do TARGET COMPILE for

6800, 6301/6801, 6809, 68000, 8080, Z80

Write or call for information on a special system to fit your requirement.

Standard systems available for these hardware —

EPSON HX-20 rom system and target compiler  
6809 rom systems for SS-50, EXORCISER, STD, ETC,  
COLOR COMPUTER  
6800 6809 FLEX or EXORCISER disk systems.  
68000 rom based systems  
68000 CP/M-68K disk systems, MODEL II/12 16

iFORTH is a refined version of FORTH Interest Group standard  
FORTH, faster than FIG-FORTH. FORTH is both a compiler and  
an interpreter. It executes orders of magnitudes faster than interpretive BASIC. MORE IMPORTANT, CODE DEVELOPMENT  
AND TESTING is much, much faster than compiled languages  
such as PASCAL and C. II Software DEVELOPMENT COSTS are  
an important concern for you, you need FORTH!

firmFORTH™ is for the programmer who needs to squeeze the  
most into roms. It is a professional programmer's tool for compact  
rommable code for controller applications.

• iFORTH and firmFORTH are trademarks of Talbot Microsystems  
• FLEX is a trademark of Technical Systems Consultants, Inc.  
• CP/M-68K is trademark of Digital Research, Inc.

iFORTH™  
from TALBOT MICROSYSTEMS  
NEW SYSTEMS FOR  
6301/6801, 6809, and 68000

#### ...> iFORTH SYSTEMS <---

For all FLEX systems: GIMIX, SWTP, SSB, or EXORCISER Specify  
5 or 8 inch diskette, hardware type, and 6800 or 6809.

• iFORTH — extended fig FORTH (1 disk) \$100 (\$15)  
with fig line editor.

• iFORTH+ — more! (3 1/2" or 2 1/2" disks) \$250 (\$25)  
adds screen editor, assembler, extended data types, utilities,  
games, and debugging aids.

• TRS-80 COLORFORTH — available from The Micro Works  
• firm FORTH — 6809 only. \$350 (\$10)

For target compilations to rommable code.  
Automatically deletes unused code. Includes HOST system  
source and target nucleus source. No royalty on targets. Re-  
quires but does not include iFORTH+.

• FORTH PROGRAMMING AIDS — elaborate decompiler \$150

• iFORTH for HX-20, in 16K roms for expansion unit or replace  
BASIC \$170

• iFORTH/68K for CP/M-68K 8" disk system \$290  
Makes Model 16 a super software development system.

• Nautilus Systems Cross Compiler  
— Requires: iFORTH+ HOST+ at least one TARGET:

— HOST system code (6809 or 68000) \$200

— TARGET source code: 6800-\$200, 6301/6801-\$200

same plus HX-20 extensions— \$300

6809-\$300, 8080 280-\$200, 68000-\$350

Manuals available separately — price in ( )  
Add \$6 system for shipping, \$15 for foreign air.

TALBOT MICROSYSTEMS 1927 Curtis Ave., Redondo Beach, CA 90278 (213) 376 9941

# AVAILABLE NOW!

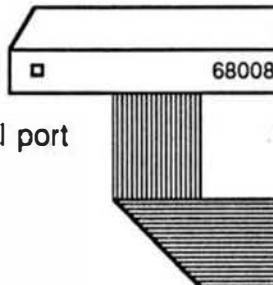
## PL $\mu$ S-68k (PL/9 for the 68000) running under FLEX<sup>TM</sup>

- Built-in screen editor
- Built-in source-level debugger
- Byte, Integer and Long variables
- Single-pass compiler
- Direct source to object
- Compiles over 1000 lines/min
- Runs on any FLEX<sup>TM</sup> system with a spare PIA port

Develop 68000 software  
on your FLEX<sup>TM</sup> system.  
No second computer  
required!

The second processor module (included):

- 10MHz 68008 CPU
- 128k bytes RAM
- Case and power supply
- Plugs into Windrush UPROM-III port



Run FLEX<sup>TM</sup>  
software on the  
68008

Other software included:

- Program loader
- 68000 FLEX<sup>TM</sup> interface package.
- Comprehensive System Monitor with source
- Hex-Binary-Hex Conversion Routines

Development is currently under way of a version for OS/9-68000. The package price includes a free copy of the OS/9 version when it becomes available.

**\$999**  
Complete

For further information, phone or write:

Worstead Laboratories  
North Walsham  
Norfolk NR28 9SA  
England

Tel (44) 692 404086  
Telex 975548 WMICRO G



# '68' MICRO JOURNAL

## OK, PLEASE ENTER MY SUBSCRIPTION

Bill My:  Mastercard  VISA   
 Card # \_\_\_\_\_ Exp. Date \_\_\_\_\_  
 For 1 Year  2 Years  3 Years

Enclosed: \$ \_\_\_\_\_

Name \_\_\_\_\_  
 Street \_\_\_\_\_  
 City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_

My Computer Is: \_\_\_\_\_

### Subscription Rates

**U.S.A.: 1 Year \$24.50, 2 Years \$42.50, 3 Years \$64.50**

**\*Foreign Surface: Add \$12.00 per Year to USA Price.**

**\*Foreign Airmail: Add \$48.00 per Year to USA Price.**

**\*Canada & Mexico: Add \$9.50 per Year to USA Price.**

**\*U.S. Currency Cash or Check Drawn on a USA Bank !**

68 Micro Journal  
 5900 Cassandra Smith Rd.  
 POB 849  
 Hixson, TN 37343



Telephone 615 842-4600

Telex 510 600-6630



CSG IMS is a general purpose information management system designed to make the development of file-intensive applications as quick and easy as possible. IMS is a full featured database manager with the added benefit of a structured, general purpose application language. Some popular applications are: accounting, inventory, data acquisition, cataloging, membership and mailing lists.

### SYSTEM FEATURES

- CSG IMS uses B+Tree index structures for fast database access and reliability. Record, index and file sizes are virtually unrestricted.
- Supported data types are: text, BCD floating point (14 digits), short and long integers, and date.
- Menu driven executive program for ease of operation.
- User definable screen forms and reports are supported.
- The interactive environment provides access to databases and most language features allowing quick ad hoc queries.
- CSG IMS includes a recursive, compiled language supporting program modules with full parameter passing.
- The CSG IMS run-time interpreter is available separately for user developed and distributed applications.
- Comprehensive 320 page manual with tutorial section.

CSG IMS for OS9/6809 LII and OS9/68000: \$495.00

Run time interpreter for CSG IMS: \$100.00

CSG IMS manual only: \$20.00

PRICES IN US DOLLARS  
 ADD \$5.00 S&H FOR CONTINENTAL USA. FOREIGN ORDERS ADD \$10.00 S&H.

### Clearbrook Software Group

To order CSG IMS or to receive further information write:

CLEARBROOK SOFTWARE GROUP  
 P.O. Box 8000-499  
 Sumas, WA 98295-8000

or phone:

(604) 853-9118

Send for a free catalog describing all of our OS9 products.  
 We welcome dealer inquiries.

OS9 is a registered trademark of Microware and Motorola.

## OS-9™ SOFTWARE

**SDISK**—Standard disk driver module allows the use of 35, 40, or 80 track double sided drives with COCO OS-9, plus you can read/write/format the OS-9 formats used by other OS-9 systems. Supports OS-9 ver. 2.0. **\$29.95**

**SDISK + BOOTFIX**—As above plus boot directly from a double sided diskette. **\$35.95**

**SK10**—Hi res (51x24) screen driver for COCO OS-9 (version 2.0 supported) **\$29.95**

**L1 UTILITY PACK**—Contains all programs formerly in Filter Kits 1 & 2, and Hacker's Kit 1 plus several additional programs. Complete "wild card" file operations, copies, moves, sorts, del, MACGEN shell command language compiler, Disassembler, Disk sector edit utility, new and improved editions, approx. 40 programs, increases your productivity. **\$49.95 (\$51.95)**

**PC-XFER UTILITIES**—Utilities to read/write and format MS-DOS diskettes on CoCo under OS-9. Also transfer files between RS disk basic and OS-9. **\$45.00** (version now available for SSB level II systems, inquire).

**CCRD 512K RAM DISK CARTRIDGE**—Requires RS Multipak Interface; OS-9 Driver and test software now included! Switch selectable address, two may be used for 1 Meg. Ramdisk. **\$199.00**

**BOLD** prices are CoCo OS-9 for OS-9 format disk, other formats (in parenthesis) specify format and OS-9 level. All orders prepaid or COD, VISA and MasterCard accepted. Add \$1.50 S&H on prepaid, COD actual charges added.

### SS-50C

#### 1 MEGABYTE RAM BOARD

Full megabyte of ram with disable options to suit any SS-50 6809 system. High reliability, can replace static ram for a fraction of the cost. **\$599** for 2 Mhz or **\$679** for 2.25 Mhz board assembled, tested and fully populated.

#### 2 MEGABYTE RAM DISK BOARD

RD2 2 megabyte dedicated ram disk board for SS-50 systems. Up to 8 boards may be used in one system. **\$1150.00**; OS-9 drivers and test program **\$30.00**.

(Add \$6 shipping and insurance, quantity discounts available.)

**D.P. Johnson, 7665 S.W. Cedarcrest St.  
Portland OR 97223 (503) 244-8152  
(For best service call between 9-11 AM Pacific Time.)**  
OS-9 is a trademark of Microware and Motorola Inc.  
MS-DOS is a trademark of Microsoft, Inc.

## COMPILER EVALUATION SERVICES

BY: Ron Anderson

The S.E. MEDIA Division of Computer Publishing Inc.  
is offering the following SUBSCRIBER SERVICE:

### COMPILER COMPARISON AND EVALUATION REPORT

Due to the constant and rapid updating and enhancement of numerous compilers, and the different utility, appeal, speed, level of communication, memory usage, etc., of different compilers, the following services are now being offered with periodic updates.

This service, with updates, will allow you who are wary or confused by the various claims of compiler vendors, an opportunity to review comparisons, comments, benchmarks, etc., concerning the many different compilers on the market, for the 6809 microcomputer. Thus the savings could far offset the small cost of this service.

Many have purchased compilers and then discovered that the particular compiler purchased either is not the most efficient for their purposes or does not contain features necessary for their application. Thus the added expense of purchasing additional compiler(s) or not being able to fully utilize the advantages of high level language compilers becomes too expensive.

The following COMPILERS are reviewed initially, more will be reviewed, compared and benchmarked as they become available to the author:

**PASCAL 'C' GSPL WHIMSICAL PL/9**

**Initial Subscription - \$39.95**  
(includes 1 year updates)  
**Updates for 1 year - \$14.50**

**S.E. MEDIA - C.P.I.  
5900 Cassandra Smith Rd.  
Hixson, TN 37343  
(615) 842-4601**

# 68000 68020 68010

# 68008 6809 6800

Write or phone for catalog.

## AAA Chicago Computer Center

120 Chestnut Lane — Wheeling IL 60090  
(312) 459-0450

Technical Consultation available most weekdays from 4 PM to 6 PM CST

68008

68000

# A Powerful 1 - 2 - 3 Combination

68010

68020

1. Stylo-Graph Word Processor  
Stylo-Merge Text Formatter  
Stylo-Spell 42,000 Word dictionary
2. Motorola 68000 Microprocessors
3. The 68K OS9 Operating System

All the Stylo programs are written in 68K assembly code making their performance second to none. The ability to always see on the screen what your printout will look like saves time and makes your work easier.

**Why settle for less than the best?  
Check it out today!**

Call or write for catalog



**Stylo Software, Inc.**

P.O. Box 916 482 C Street  
IDAHO FALLS, IDAHO 83402  
(208) 529-3210

VISA OR MASTERCARD ACCEPTED

## DATA-COMP SPECIAL Heavy Duty Power Supplies

For a limited time we are offering our HEAVY DUTY SWITCHING POWER SUPPLY. These are BRAND NEW units and will not last long. Also note that these prices are less than 1/4 the normal price for these high quality units.



A Division of  
Computer Publishing, Inc.  
5900 Cassandra Smith Road  
Hixson, TN 37343  
Telephone 615 842-4600  
Telex 510 600-6630



Make: Boschert

Size: 10.5 x 5 x 2.5 inches - including heavy mounting bracket and baseplate.

Rating: in 110/220 volts ac (strap change) Out: 130 watts

Output: +5v - 10 amps  
+12v - 4.0 amps  
+12v - 2.0 amps  
-12v - 0.5 amps

Mating Connector: Terminal strip  
Load Reaction: Automatic short circuit recovery

Each  
**SPECIAL: \$59.95**  
**2 or more 49.95**

Add: \$7.50 each S/H

Make: Boschert

Size: 10.75 x 6.2 x 2.25 inches

Rating: 110/220 ac (strap change) Out: 81 watts

Output: +5v - 8.0 amps  
+12v - 2.4 amps  
+12v - 2.4 amps  
+12v - 2.1 amps  
-12v - 0.4 amps

Mating Connectors: Molex  
Load Reaction: Automatic short circuit recovery

Each  
**SPECIAL: \$49.95**  
**2 OR MORE 39.95**

Add: \$7.50 S/H each

# 6809<>68XXX UniFLEX

## X-TALK A C-MODEM/Hardware Hookup

Exclusive for the MUSTANG-020 running UniFLEX, is a new transfer program and cable set from DATA-COMP (CPI). X-TALK consist of 2 disks and a special cable, this hook-up enables a 6809 SWTPC UniFLEX computer to port UniFLEX files directly to a 68XXX UniFLEX system.

This is the only currently available method to transfer files, text or otherwise, from a 6809 UniFLEX system to a 68000 UniFLEX system, that we have seen. A must if you want to recompile or cross assemble your old (and valuable) source files to run on a 68000 UniFLEX system. GIMIX users can directly transfer files between a 6809 GIMIX system and our MUSTANG-020 68020 system, or GIMIX 68020 system. All SWTPC users must use some sort of method other than direct disk transfer. The 6809 SWTPC UniFLEX disk format is not readable by most other 68000 type systems.

The cable is specially prepared with internal connections to match the non-standard SWTPC SO/9 DB25 connectors. A special SWTPC+ cable and software is also available, at the same price. Orders must specify which type SWTPC 6809 UniFLEX system they intend to transfer from or to.

The X-TALK software is furnished on two disks. One 8" disk containing the 6809 software and one 5" disk containing the 68XXX software. These programs are also complete MODEM programs and can be used as such, including X-on X-off, and all the other features you would expect from a full modem program.

X-TALK can be purchased with/without the special cables, however, this SPECIAL price is available only to registered MUSTANG-020 owners.

**X-TALK, w/cable \$99.95  
X-TALK only 69.95  
X-TALK w/source \$149.95**

**DATA-COMP**  
5900 Cassandra Smith Rd.  
Hixson, TN 37343

Telephone 615 842-4601  
Telex 510 600-6630

Note: Registered MUSTANG-020 owners must furnish system serial number in order to buy at these special low prices.

## 68' MICRO JOURNAL

Disk-1 Fiction, Musical, Microcopy, Minima, "Lifetime," "Poetry," "Foodies," "Disk.  
Disk-2 Diskedit w/ net & tools, Prime, "Primad," "Snoopy," "Football," "Hazepon," "Lifetime  
Disk-3 Cbugp09, Sec1, Sec2, Find, Table2, Intext, Disk-exp, "Disksave.  
Disk-4 Mailing Program, "Findas," "Change," "Textdisk.  
Disk-5 DISKFDX 1, "DISKFDX 2, "LETTER, "LOVDESIGN,  
"BLACKJAK, "BOWLING.  
Disk-6 "Purchase Order, Index (Disk file index)  
Disk-7 Linking Loader, Rload, Hartress  
Disk-8 Crest, Lanpher (May 82)  
Disk-9 Diskcopy, Diskedit (Aug 82)  
Disk-10 Home Accounting (July 82)  
Disk-11 Dissembler (June 84)  
Disk-12 Modem68 (May 84)  
Disk-13 Interm68, Testm68, "Cleanup," "Diskalign," Help,  
Date.Txt  
Disk-14 "Int, "Test," Terminal, "Find, "Diskedit, Int.Lib  
Disk-15 Modem9 + Updates (Dec. 84 Gilchrist) to Modem9  
(April 84 Comm)  
Disk-16 Copy.Txt, Copy.Doc, Cat.Txt, Cat.Doc  
Disk-17 Match Utility, RATBAS, A Basic Preprocessor  
Disk-18 Parse.Mod, Size.Cmd (Sept. 85 Armstrong),  
CMDCODE, CMD.Txt (Sept. 85 Spray)  
Disk-19 Clock, Date, Copy, Cat, PDEL\_Asm & Doc., Errors.Sys,  
Do, Log.Asm & Doc.  
Disk-20 UNDI Like Tools (July & Sept. 85 Taylor & Gilchrist).  
Dragon.C, Grep.C, LS.C, FDUMP.C  
Disk-21 Utilities & Games - Data, Lite, Madness, Touch, Goblin,  
Starbit, & 15 more.  
Disk-22 Read CPM & Non-FLEX Disks, Fraser May 1984.  
Disk-23 ISAM, Indexed Sequential file Accessing Methods,  
Condon Nov. 1985, Extensible Table Driven Language  
Recognition Utility, Anderson March 1986.  
Disk-24 68' Micro Journal Index of Articles & B4 Bucket Items  
from 1979 - 1985, John Current.  
Disk-25 KERMIT for FLEX derived from the UNDX ver. Burg  
Feb. 1986, (2)-5" Disks or (1)-8" Disk.  
Disk-26 Compact UnBoard Review, Code & Diagram, Burison  
March '86.  
Disk-27 ROTABIT.TXT, SUMTEST.TXT, CONDATA.TXT,  
BADMEM.TXT  
Disk-28 CT-82 emulator, bit mapped.  
Disk-29 "StarTrek

### NOTE:

This is a reader service ONLY! No warranty is offered or implied, they are as received by '68' Micro Journal, and are for reader convenience ONLY (some MAY include fixes or patches). Also 6800 and 6809 programs are mixed, as each is fairly simple (mostly) to convert to the other.

**8" Disk \$14.95      5" Disk \$12.95**

**68' Micro Journal**  
5900 Cassandra Smith Rd. Hixson, TN 37343

**(615)-842-4600**

**Telex 5106006630**

\*Indicates 6800  
\*\*Indicates BASIC SWTPC or TSC  
6809 no Indicator  
Foreign Orders Add \$4.50 for Surface Mail  
or \$7.00 for Air Mail  
\*All Currency in U.S. Dollars



# 6809/68008 SINGLE BOARD COMPUTERS

The Peripheral Technology Family of Single Board Computers is a Low-Cost Group Which Ranges From an Entry Level 8-Bit Version to a Powerful 68008-Based Board. A Product is Available to Fit Almost Every User's Requirements.

## PT69-5

- 6809 Processor/2MHz Clock
- 4 RS-232 Serial Ports
- 2 8-Bit Parallel Ports
- 4K-16K EPROM/60K Ram
- Parallel Printer Interface
- DS/DD Controller for 35-80
- Track Drives Ranging From SS/SD-DS/DD
- Winchester Interface Port

PRICE: \$349.00



## PT69-3

- 6809 1 MHz Processor
- 2 RS-232 Serial Ports
- 2 8-Bit Parallel Ports
- 4K EPROM/59K User Ram
- DS/DD Controller for 35-80 Track Drives Ranging From SS/SD-DS/DD

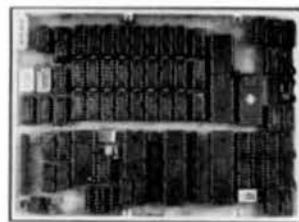
PRICE: \$269.95

OS9 L1 For  
PT69 BOARDS: \$200.00  
SK'DOS: \$ 49.95

## PT68K-1

- MC68008 10 MHz Processor
- 768K RAM/64K EPROM
- 2 RS-232 Serial Ports
- Winchester Interface Port
- Floppy Disk Controller for 2 5 1/4" Drives
- 2 8-Bit Parallel Ports

BOARD: \$595.00  
WITH OS9: \$749.95  
WITH SK'DOS: \$675.00



**PERIPHERAL TECHNOLOGY**  
1480 Terrell Mill Road, Suite 870  
Marietta, Georgia 30067  
(404) 984-0742 Telex # 880584  
VISA/MASTERCARD/CHECK/C.O.D.

Send For Catalogue For Complete Information On All Products.

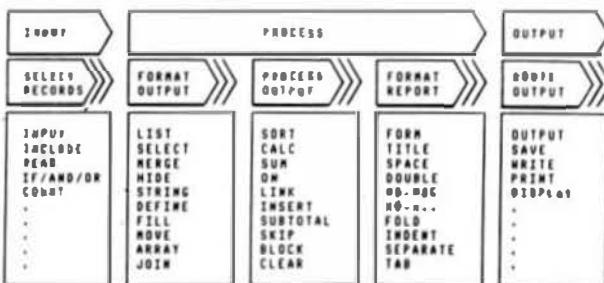
<sup>TM</sup>OS9 is A Trademark Of Microware and Motorola.

# XDMS-IV

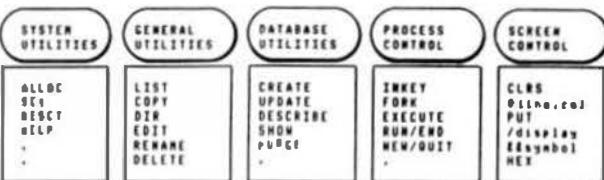
## Data Management System



### XDMS-IV PROCESSING MODEL



### XDMS-IV SUPPORT COMMANDS



Up to 32 groups/fields per record! Up to 12 character field names! Up to 1024 byte records! Input-Process-Output (IPO) command structure! Upper/Lower case commands! User defined screen and print control! Process files! Form files! Conditional execution! Process chaining! Upward/Downward file linking! File joining! Random file virtual paging! Built in utilities! Built in text line editor! Fully session oriented! Enhanced forms! Boldface, Double width, Italics and Underline supported! Write in compact structured assembler! Integrated for FAST execution!

### XDMS-IV Data Management System

XDMS-IV is a brand new approach to data management. It not only permits users to describe, enter and retrieve data, but also to process entire files producing customized reports, screen displays and file output. Processing can consist of any of a set of standard high level functions including record and field selection, sorting and aggregation, lookups in other files, special processing of record subsets, custom report formatting, totaling and subtotaling, and presentation of up to three related files as a "database" on user defined output reports.

### POWERFUL COMMANDS:

XDMS-IV combines the functionality of many popular DBMS software systems with a new easy to use command set into a single integrated package. We've included many new features and commands including a set of general file utilities. The processing commands are Input-Process-Output (IPO) oriented which allows almost instant implementation of a process design.

### SESSION ORIENTED:

XDMS-IV is session oriented. Enter "XDMS" and you are in instant command of all the features. No more waiting for a command to load in from disk! Many commands are immediate, such as CREATE (file definition), UPDATE (file editor), PURGE and DELETE (utilities). Others are process commands which are used to create a user process which is executed with a RUN command. Either may be entered into a "process" file which is executed by an EXECUTE statement. Processes may execute other processes, or themselves, either conditionally or unconditionally. Menus and screen prompts are easily coded, and entire user applications can be run without ever leaving XDMS-IV!

### IT'S EASY TO USE!

XDMS-IV keeps data management simple! Rather than design a complex DBMS which hides the true nature of the data, we kept XDMS-IV file oriented. The user view of data relationships is presented in reports and screen output, while the actual data resides in easy to maintain files. This aspect permits customized presentation and reports without complex redefinition of the database files and structure. XDMS-IV may be used for a wide range of applications from simple record management systems (addresses, inventory...) to integrated database systems (order entry, accounting...). The possibilities are unlimited...

XDMS-IV for 6809 PLX, STAR-DOS, 82+DOS 10" or 8".....\$350.00+P&H  
Order by Phone! 615-462-4600/4601 - VISA and MasterCard accepted!  
Or write South East Media, 3900 Cassandra Smith, Kinston, Tenn 37343

WESTCHESTER Applied Business Systems  
2 Fox Pond Lane, Briarcliff Manor, N.Y. 10510 Tel 914-941-2552(Bus)  
FLX(8080) (m) STAR-CITE Corp.





Now Offering: "FLEX" (2 Versions)  
AND "STAR-DOS PLUS+"™

A Family of 100% 68XX Support Facilities  
The Folks who FIRST Put FLEX™ on  
The CoCo

**FLEX-CoCo Sr.**  
with TSC Editor  
TSC Assembler  
Complete with Manuals  
Reg. '250. Only '79."

#### STAR-DOS PLUS+

- Functions Same as FLEX
- Reads - writes FLEX Disks '34.™
- Run FLEX Programs
- Just type: Run "STAR-DOS"
- Over 300 utilities & programs to choose from.

**FLEX-CoCo Jr.**  
without TSC  
Editor & Assembler  
'49.™

#### PLUS

#### ALL VERSIONS OF FLEX & STAR-DOS INCLUDE

**TSC Editor**  
Reg \$50.00  
**NOW \$35.00**

- + Read-Write-Dir RS Disk
- + Run RS Basic from Both
- + More Free Utilities

- + External Terminal Program
- + Test Disk Program
- + Disk Examine & Repair Program
- + Memory Examine Program
- + Many Many More!!!

**TSC Assembler**  
Reg \$50.00  
**NOW \$35.00**

**CoCo Disk Drive Systems**

2 THINLINE DOUBLE SIDED DOUBLE DENSITY DISK DRIVES SYSTEM WITH POWER SUPPLY, CABINET, DISK DRIVE CABLE, J6M NEW DISK CONTROLLER JPD-CP WITH J-DOS, RS-DOS OPERATING SYSTEMS. \$469.95

\* Specify What CONTROLLER You Want J&M, or RADIO SHACK

THINLINE DOUBLE SIDED DOUBLE DENSITY 40 TRACKS \$129.95

#### MISC

64K UPGRADE FOR C,D,E,F, AND COCO II	\$ 29.95
RADIO SHACK BASIC 1.2	\$ 24.95
RADIO SHACK DISK BASIC 1.1	\$ 24.95

DISK DRIVE CABINET FOR A SINGLE DRIVE	\$ 49.95
DISK DRIVE CABINET FOR TWO THINLINE DRIVES	\$ 69.95

#### Verbatim Diskettes

Single Sided Double Density	\$ 24.00
Double Sided Double Density	\$ 24.00

#### PRINTERS

EPSON LX-80	\$289.95
EPSON MX-70	\$125.95
EPSON MX-100	\$495.95

#### Controllers

J&M JPD-CP WITH J-DOS WITH J-DDS, RS-DOS	\$139.95
RADIO SHACK 1.1	\$159.95
RADIO SHACK 1.1	\$134.95

#### ACCESSORIES FOR EPSON

BJ48 2K SERIAL BOARD	\$ 89.95
BJ49 32K EXPAND TO 128K	\$165.95
EPSON MX-RX-80 KIBBONS	\$ 7.95
EPSON LX-80 BIBBONS	\$ 5.95
TEAC/TEC UNITS FOR LX-80 CABLES & OTHER INTERFACES	\$ 39.95
CALL FOR PRICING	

#### Disk Drive Cables

Cable for One Drive	\$ 19.95
Cable for Two Drives	\$ 24.95

**DATA-COMP**  
5900 Cassandra Smith Rd.  
Hixson, TN 37343



**SHIPPING**  
USA ADD 2%  
FOREIGN ADD 5%  
MIN. \$2.50

**(615) 842-4600**  
For Ordering  
Telex 5106006630

# Introducing.....

## S - 50 BUS / 68XX

Board and/or Computer  
Terminals-CRTs-Printers  
Disk Drives-etc.

## REPAIRS

### NOW AVAILABLE TO ALL S50/68XX USERS

The Data-Comp Division of CPI is proud to announce the availability of their service department facilities to 'ALL' S50 Bus and 68XX users. Including all brands, SWTPC - GiMIX - SSB - HELIX and others, including the single board computers. \*Please note that kit-built components are a special case, and will be handled on an individual basis, if accepted.

1. If you require service, the first thing you need to do is call the number below and describe your problem and confirm a *Data-Comp service & shipping number*. This is very important, Data-Comp will not accept or repair items not displaying this number! Also we cannot advise or help you troubleshoot on the telephone, we can give you a shipping number, but NO advice! Sorry!

2. All service shipments must include both a minimum \$40.00 estimate/repair charge and pre-paid return shipping charges (should be same amount you pay to ship to Data-Comp).

3. If you desire a telephone estimate after your repair item is received, include an additional \$5.00 to cover long distance charges. Otherwise an estimate will be mailed to you, if you requested an estimate. Estimates must be requested. Mailed estimates slow down the process considerably. However, if repairs are not desired, after the estimate is given, the \$40.00 shall constitute the estimate charge, and the item(s) will be returned unrepaird providing sufficient return shipping charges were included with the item to be serviced. Please note that estimates are given in dollar amounts only.

4. Data-Comp service is the oldest and most experienced general S50/68XX service department in the world. We have over \$100,000.00 in parts in stock. We have the most complete set of service documents for the various S50/68XX systems of anyone - YET, WE DO NOT HAVE EVERYTHING! But we sure have more than anyone else. We repair about 90% of all items we receive. Call for additional information or shipping instructions.

**DATA-COMP**  
5900 Cassandra Smith Rd.  
Hixson, TN 37343

(615)842-4607  
Telex 5106006630



↑  
**This**



**Not This** ↓

